

文章编号: 2095-2163(2023)03-0001-10

中图分类号: TP929.5

文献标志码: A

# 改进型 DV-Hop 定位模型及其求解的混合视觉进化神经网络

郑悦<sup>1</sup>, 张著洪<sup>1,2</sup>

(1 贵州大学 大数据与信息工程学院, 贵阳 550025; 2 贵州大学 贵州省系统优化与科学计算特色重点实验室, 贵阳 550025)

**摘要:** 针对 DV-Hop 定位模型跳距估计精度低且难准确获取未知节点位置的问题, 提出改进型 DV-Hop 定位优化模型及其求解的混合视觉进化神经网络优化算法。DV-Hop 定位模型设计中, 引入 4 种通信半径细化节点间的跳数计算, 并利用动态权重因子修正最小跳数及平均跳距计算模型, 获得改进型 DV-Hop 的计算模型。算法设计中, 依据果蝇和蝗虫视觉系统的信息处理机制, 建立输出全局和局部学习率的混合视觉神经网络, 进而在此神经网络的输出引导下, 借助哈里斯鹰优化算法中哈里斯鹰的位置更新策略设计状态更新策略, 由此获得能求解 DV-Hop 定位问题的视觉进化神经网络优化算法。比较性的数值实验显示, 该算法求解基准函数优化问题具有明显优势, 并对未知节点的定位精度高且收敛速度快。

**关键词:** DV-Hop; 混合视觉进化神经网络; 哈里斯鹰优化; 状态更新

## Improved DV-Hop localization model and related hybrid visual neural network

ZHENG Yue<sup>1</sup>, ZHANG Zhuhong<sup>1,2</sup>

(1 College of Big Data and Information Engineering, Guizhou University, Guiyang 550025, China; 2 Guizhou Provincial Characteristic Key Laboratory of System Optimization and Scientific Computation, Guizhou University, Guiyang 550025, China)

**[Abstract]** This work develops an improved DV-Hop optimization model and the related optimization approach, in order to solve the problems of the low accuracy of jump distance estimation in the DV-Hop positioning algorithm and the difficulty of accurately obtaining the positions of unknown nodes. In the design of the model, four kinds of communication radii are introduced to calculate the hop numbers between nodes, and later, the original minimum hop number and average hop distance models are improved in terms of dynamic weight factors. In the design of the algorithm, based on the visual information-processing mechanism of the fruit fly and locust visual systems, a hybrid visual neural network, which can generate global and local learning rates, is developed and integrated with a Harris Hawk optimization-based state update strategy to construct a hybrid visual neural network optimization approach. The comparative experiments have validated that, when solving the benchmark function optimization problems, the algorithm is of significant advantage over the compared approaches and can locate the unknown nodes with high accuracy and rapid convergence.

**[Key words]** DV-Hop; hybrid visual neural network; Harris hawk optimization; state update

## 0 引言

无线传感器网络(Wireless sensor network, WSN)是一种分布式自组织网络,是通过节点间相互协作实现对指定监控对象的数据感知、采集和处理,并将数据传输给用户;因其节点具有体积小、成本低、布设灵活及功耗低等特点<sup>[1]</sup>,使得传感器在诸如环境监测、医疗、军事、智能家居等领域中具有广阔发展前景<sup>[2]</sup>。WSN 定位包括测距定位和非测距定位<sup>[3]</sup>。前者需通过测量节点间的实际距离或利用方位角进行准确定

位,但硬件投入高。相反,后者依据网络的连通性进行定位,其成本低、功耗小、抗干扰性能好,且实用性强。距离矢量跳数(Distance Vector-Hop, DV-Hop)节点定位算法是一种基于 GPS 定位的非测距定位方法,也是目前 WSN 中应用最为广泛的算法之一<sup>[4]</sup>。由于其节点定位精度主要依赖 WSN 的连通度完成对未知节点的定位,加之初始节点的分布具有随机性以及采用最小二乘法或数值迭代法估计未知节点的位置<sup>[5-7]</sup>,导致节点的定位性能有待提升。例如, Kaur 等学者<sup>[5]</sup>利用高斯-牛顿法估计未知节点的位置,能

基金项目: 国家自然科学基金(62063002)。

作者简介: 郑悦(1997-),女,硕士研究生,主要研究方向:智能优化算法;张著洪(1966-),男,博士,教授,博士生导师,主要研究方向:数据科学与计算智能、深度学习等。

通讯作者: 张著洪 Email: zhzhang@gzu.edu.cn

收稿日期: 2022-10-14

够提高未知节点的定位精度,但计算复杂度高。张毓等学者<sup>[6]</sup>基于 RSSI 与加权质心定位,将已被定位的未知节点转化为锚节点,进而对剩余节点定位,得到的算法能有效解决因节点连通度小而出现定位精度低的问题,但定位效果受环境干扰因素的影响较大。Ma 等学者<sup>[7]</sup>提出一种基于多通信半径的定位算法,其通过锚节点多次广播,计算锚节点与未知节点间的最小跳数,但未涉及未知节点间的最小跳数计算问题。另外,智能优化算法已初步被用于未知节点的定位<sup>[8-9]</sup>。例如,褚银菲等学者<sup>[8]</sup>利用改进型差分进化算法对未知节点进行定位,相较于 DV-Hop 定位算法,能较大地提高定位精度,但性能受参数的影响较大,且种群多样性易于过早消失。

另一方面,视觉神经网络作为一种模拟视觉神经元响应特性而建立的前馈神经网络已得到可观研究进展,迄今为止则应用于机器人导航、目标跟踪、碰撞检测等方面<sup>[10-12]</sup>。Fu 等学者<sup>[10]</sup>利用果蝇与蝗虫两种生物的视觉响应特性,提出一种混合神经网络模型,并已用于机器人的运动跟踪与导航。另外,本文研究团队围绕视觉运动检测和最优化问题的求解,从视觉神经网络模型研究及应用角度已展开一些研究;例如,针对平移、旋转等运动模式的识别,利用蝗虫与猕猴等生物中的视觉响应特性提出人工蝗虫视觉神经网络<sup>[13]</sup>;基于视觉信息处理机制与种群进化或梯度下降思想,获得可调参数少且寻优精度高的视觉进化、学习神经网络<sup>[14-16]</sup>,并将其应用于 RSSI 定位、复杂高维优化问题的求解,但算法的定位精度及搜索效果的稳定性有待提高。

综上,基于现有求解 DV-Hop 节点定位问题的研究现状,本文在建立改进型节点跳数、平均跳距计算模型基础上,基于果蝇视觉、蝗虫视觉信息处理机制及哈里斯鹰优化的种群更新策略,提出一种混合视觉进化神经网络(Hybrid visual evolutionary neural network, HVENN),并用于 DV-Hop 的节点定位。

## 1 改进型 DV-Hop 模型与求解准备

### 1.1 DV-Hop 的距离计算

假定 WSN 由  $m$  个锚节点和  $n$  个未知节点构成,锚节点  $i$  和未知节点  $j$  的位置分别为  $A_i(x_i, y_i)$  及  $O_j(x_j, y_j)$ ,  $1 \leq i \leq m$ ,  $1 \leq j \leq n$ 。基本 DV-Hop 定位方法经由锚节点与未知节点间的最小跳数以及锚节点的平均跳距计算未知节点到锚节点的距离。具体如下:

首先,WSN 网络中一个锚节点(例如节点  $A$ )向

所有其它锚节点和未知节点均发送一个包含自身位置、跳数和地址的包,跳数初始值为 0;然后,距离节点  $A$  最近的锚节点(例如节点  $B$ )向剩余的锚节点和未知节点也发送一个同类型的包。如此继续,直到所有锚节点均发送自身的包为止。之后,经由 DV-Hop 的节点跳数计算方法<sup>[4]</sup>,可获 WSN 中锚节点  $A_i$  到锚节点  $A_j$  的最小跳数  $hop_{ij}$  及到未知节点  $O_j$  的最小跳数  $Hop_{ij}$ 。其次,节点  $A_i$  依据式(1)得到自身的平均跳距:

$$Ahop_i = \frac{\sum_{1 \leq j \leq m, j \neq i} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{\sum_{1 \leq j \leq m, j \neq i} hop_{ij}} \quad (1)$$

进而,经由式(2)计算  $O_j$  到  $A_i$  的距离:

$$d_{ji} = Ahop_i \times Hop_{ij} \quad (2)$$

### 1.2 改进型节点定位优化模型

由于未知节点的位置具有不确定性,加之节点的位置分布不均匀,导致 DV-Hop 定位算法存在以下缺陷:

(1) 在计算锚节点与未知节点的距离时,通过把节点间的连线当作直线来计算最小跳数值,导致因传感器节点的分布不均匀,跳数的计算存在误差。

(2) 锚节点与未知节点间的距离是通过节点间的平均跳距而获得,进而因平均跳距、跳数的计算误差,使得未知节点的定位精度受到极大影响。

针对 DV-Hop 定位存在的以上缺陷,这里对锚节点到锚节点和锚节点到未知节点的最小跳数( $hop_{ij}$ 、 $Hop_{ij}$ )以及平均跳距计算模型做如下改进:

(1) 改进型最小跳数计算模型。假设锚节点  $A$  的通信半径为  $R$ ,以其为圆心, $R$ 、 $3R/4$ 、 $R/2$ 、 $R/4$  为半径产生 4 个同心圆,如图 1 所示。锚节点  $A$  以  $R/4$  为半径向网络中泛洪广播数据,接收到节点  $A$  信号的未知节点首先查询是否接收到过该节点的跳数;若有,则维持之前跳数信息不变;否则,将未知节点与该节点  $A$  间的跳数记为  $1/4$  跳。以此类推,直到节点  $A$  以  $R$  为半径向网络中泛洪广播数据为止。

若节点  $A_i$  与  $O_j$  的实际距离为  $d$ ,则最小跳数  $Hop_{ij}$  经由式(3)确定:

$$Hop_{ij} = \begin{cases} \frac{1}{4} & 0 < d \leq \frac{R}{4} \\ \frac{1}{2} & \frac{R}{4} < d \leq \frac{R}{2} \\ \frac{3}{4} & \frac{R}{2} < d \leq \frac{3R}{4} \\ 1 & \frac{3R}{4} < d \leq R \end{cases} \quad (3)$$

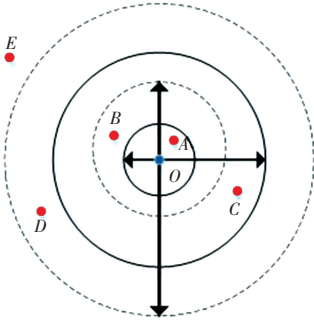


图1 4种通信半径下的广播跳数

Fig. 1 Number of broadcast hops at four communication radii

其次,在锚节点  $A_i$  与未知节点间引入如下加权因子:

$$W_i = \frac{1}{\sum_{j=1}^n \frac{1}{hop_{ij}}} \quad (4)$$

获得节点  $A_i$  到  $O_j$  的如下改进型最小跳数计算模型:

$$NHop_{ij} = \sum_{j=1}^n W_i Hop_{ij} \quad (5)$$

(2) 改进型平均跳距计算模型。获得锚节点  $A_i$  与  $A_j$  间的最小跳数  $hop_{ij}$  后,经由式(6)计算锚节点  $A_i$  的平均跳距:

$$Ahop_i = \frac{\sum_{i \neq j} hop_{ij} \times d_{ij}}{\sum_{i \neq j} hop_{ij}^2} \quad (6)$$

在此,  $d_{ij}$  为锚节点  $A_i$  到  $A_j$  的距离。进而,基于二者的最小跳数  $hop_{ij}$  和式(6),获得二者间的距离估计模型:

$$\hat{d}_{ij} = Ahop_i \times hop_{ij} \quad (7)$$

另一方面,为进一步减小跳数计算误差,利用锚节点间的实际距离与估计距离误差修正  $Ahop_i$ , 即:

$$NAhop_i = \sum_{i=1}^m \varepsilon_i Ahop_i \quad (8)$$

$$\varepsilon_i = \frac{\sum_{i \neq j} (\hat{d}_{ij} - d_{ij})}{\sum_{i \neq j} hop_{ij}} \quad (9)$$

其中,  $\varepsilon_i$  为锚节点  $i$  的平均每跳误差。最后,由该跳距和跳数求出未知节点  $O_j$  到锚节点  $A_i$  的距离,即:

$$d_{ji} = NAhop_i \times NHop_{ij} \quad (10)$$

结合式(10),  $n$  个未知节点的位置估计问题可通过求解如下 DV-Hop 模型 (DV-Hop model, DV-Hop-M) 加以解决:

$$\min f(x) = \sum_{i=1}^m \sum_{j=1}^n \left| \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} - d_{ji} \right| \quad (11)$$

其中,  $x$  表示  $n$  个未知节点的位置坐标构成的决策向量。

### 1.3 模型求解准备

考虑如下最优化问题:

$$\min_{x \in D} f(x) \quad (12)$$

其中,  $D$  为  $p$  维欧氏空间中有界闭区域;  $f(x)$  为连续型函数;称  $x^*$  为最优解,若对于  $D$  中任意候选解或称为状态的  $x$ ,均有  $f(x^*) \leq f(x)$ 。取  $M \times N$  个状态构成抽象的状态矩阵  $A$ , 即  $A = (x_{ij})_{M \times N}$ ; 将状态  $x$  对应的目标函数值  $f(x)$  视为灰度值或光强度,  $A$  中各状态对应的灰度值构成的灰度图记作  $f(A)$ 。第  $t$  时刻的状态矩阵  $A^{(t)}$  对应的灰度图记为  $f(A^{(t)})$ 。在此,  $A^{(t)} = (x_{ij}^{(t)})_{M \times N}$ , 相应的灰度图为  $f(A^{(t)})$ 。以  $f(A^{(t)})$  作为视觉输入图像,借助果蝇、蝗虫视觉系统的信息处理机制,在已有神经网络<sup>[17-18]</sup>基础上,建立混合型视觉神经网络 (Hybrid visual neural network, HVNN)。随后,借助此神经网络输出的局部行为量  $\lambda_{ij}(t)$ 、全局行为量  $\rho(t)$ , 以及哈里斯鹰优化的种群更新策略,经由式(13)转移状态矩阵  $A^{(t)}$  中状态  $x_{ij}^{(t)}$  为  $x_{ij}^{(t+1)}$ :

$$x_{ij}^{(t+1)} = x_{ij}^{(t)} + \Delta(x_{ij}^{(t)}, \lambda_{ij}(t), \rho(t)), \quad 1 \leq i \leq M, 1 \leq j \leq N \quad (13)$$

在此,  $\lambda_{ij}(t)$  和  $\rho(t)$  作为调节因子引导状态的转移。

## 2 混合视觉进化神经网络

### 2.1 混合视觉神经网络

人工果蝇视觉神经网络 (Artificial fly visual neural network, AFVNN)<sup>[17]</sup> 与蝗虫视觉神经网络 (Locust visual neural network, LVNN)<sup>[18]</sup> 是分别源于果蝇、蝗虫视觉信息处理机制的前馈神经网络。AFVNN 由 4 个视觉信息处理层 (光感受器 (Photoreceptor)、视网膜 (Retina)、薄膜 (Lamina)、髓质 (Medulla)) 和一个小叶板切向细胞 (Lobula plate tangential cell, LPTC) 构成;在此,光感受器接收视觉环境中的光亮强度;视网膜将光感受器获取的光亮强度经平滑处理,提取运动目标;薄膜层经侧抑制机制产生刻画运动目标变化状态的兴奋量;髓质层输出运动目标的局部方向行为量;LPTC 汇集所获局部方向行为量,并产生刻画运动目标变化状态的全局方向行为量。LVNN 由光感受器层



(Photoreceptor)、兴奋层(Excitation)、抑制层(Inhibition)、兴奋抑制合成层(Stimulation)和一个蝗虫视觉神经元构成。在此,兴奋层和抑制层接收当前及上一时刻的灰度图;兴奋抑制合成层汇聚此2层输出的行为量,并经由侧抑制获取运动目标的局部方向行为量;蝗虫视觉神经元聚合此局部方向行为量,并产生刻画运动目标变化状态的全局运动方向行为量。AFVNN和LVNN的视觉神经生理理论 and 设计原理可由文献<sup>[17-18]</sup>获知。由于果蝇与蝗虫的视觉系统均具有特征提取和运动方向检测神经元,所以如此2种神经网络均能输出刻画运动目标逼近、偏离摄像头的全局运动方向量。

HVNN由AFVNN中的Retina、Lamina、Medulla、LPTC以及LVNN中的Inhibition、Stimulation层构成,其拓扑结构如图2所示。各层的设计如下:

(1) Retina: 由 $M \times N$ 个Retina节点以矩阵形式排列而成,节点 $(i, j)$ 在 $t$ 时刻接收 $\mathbf{A}^{(t)}$ 中状态 $\mathbf{x}_{ij}^{(t)}$ 对应的光强度 $f(\mathbf{x}_{ij}^{(t)})$ ,并经由式(14)生成光强度偏差量:

$$e_{ij}(t) = f(\mathbf{x}_{ij}^{(t-1)}) - f(\mathbf{x}_{ij}^{(t)}) \quad (14)$$

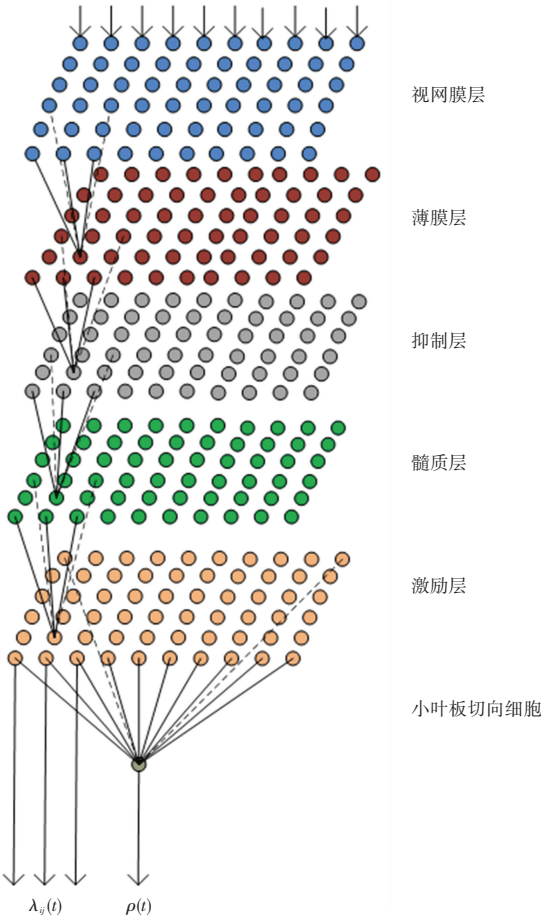


图2 混合视觉神经网络

Fig. 2 Hybrid visual neural network

若节点 $(i, j)$ 处的状态未被激活,则经由以下扰动策略激活:

$$y_{ij}(t) = \begin{cases} \mathbf{x}_{ij}^{(t)} + \frac{f(\mathbf{x}_{ij}^{(t)}) - \min}{\max - \min} (1 - \frac{t}{G_{\max}}) \boldsymbol{\zeta} & \text{if } e_{ij}(t) = 0 \\ \mathbf{x}_{ij}^{(t)} & \text{else} \end{cases} \quad (15)$$

其中, $\boldsymbol{\zeta}$ 是每个分量在0与1之间服从均匀分布的随机向量; $\max$ 与 $\min$ 分别是 $\mathbf{A}^{(t)}$ 中所有状态对应的灰度值中最大和最小值; $G_{\max}$ 为最大时间步。记 $\mathbf{B}^{(t)} = (\mathbf{y}_{ij}(t))_{M \times N}$ 。

于是,节点 $(i, j)$ 的光强度偏差量由式(16)确定:

$$m_{ij}(t) = \begin{cases} e_{ij}(t) & e_{ij} \neq 0 \\ f(\mathbf{y}_{ij}(t)) - f(\mathbf{x}_{ij}^{(t)}) & \text{else} \end{cases} \quad (16)$$

(2) Lamina: 由果蝇的cartridge(cart)和蝗虫的Inhibition(I)层构成,每个子层的节点数均为 $M \times N$ 。cartridge层经由对Retina层做扩边处理后, cart节点 $(i, j)$ 将接收Retina层中对应节点处 $3 \times 3$ 邻域内节点的光强度偏差量,并经由高斯卷积获输出行为量,即:

$$cart_{ij}(t) = \frac{1}{w} \sum_{0 \leq k, l \leq 2} w_{kl} m_{i+k-1, j+l-1}(t) \quad k, l \neq 1 \quad (17)$$

其中, $w$ 是下列权重矩阵的元素之和:

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

这里, $w_{kl}$ 是 $\mathbf{W}$ 中位置 $(k, l)$ 处的元素。

Inhibition层对cartridge层做扩边处理后,节点 $(i, j)$ 接收来自cartridge层中对应节点的 $3 \times 3$ 邻域内节点的行为量,并经由以下侧抑制策略获得该节点的抑制量<sup>[15]</sup>:

$$I_{ij}(t) = \sum_{i=-1}^1 \sum_{j=-1}^1 cart(i, j) w_l(i, j) \quad (18)$$

其中, $(i, j) \neq (0, 0)$ ,  $w_l(i, j)$ 是局部抑制权重,即下列权重矩阵中位置 $(i, j)$ 处的元素:

$$\mathbf{w}_l = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

(3) Medulla: 此层由 $m_1, m_2$ 两个节点子层构成,且各层的规模均为 $M \times N$ 。 $m_1$ 节点 $(i, j)$ 首先接收Inhibition层中节点 $(i, j)$ 的 $3 \times 3$ 邻域内节点的输出,然后基于单向EMD运动方向检测器<sup>[19]</sup>,获得输出行为量:

$$m_1(i, j | i+k, j+l) = I_{i+k, j+l}(t)g(I_{ij}(t)) - I(i, j)g(I_{i+k, j+l}(t)) \quad -1 \leq k, l \leq 1, k, l \neq 0 \quad (19)$$

随后,  $m_2$  节点  $(i, j)$  经由与之匹配的  $m_1$  节点处  $3 \times 3$  邻域内节点的方向量, 以及 AFVNN 中<sup>[17]</sup> Medulla 的设计, 获得该节点在水平、竖直方向的方向量  $m_{2h}(i, j)$  及  $m_{2v}(i, j)$ 。

(4) Stimulation: 由  $M \times N$  个节点以矩阵形式排列而成, 节点  $(i, j)$  接收  $m_2$  节点  $(i, j)$  在水平、竖直方向的输出行为量, 进而通过改进文献<sup>[12]</sup> 中激励层的设计, 经由式(20) 获得该节点在水平和竖直方向的行为量:

$$\lambda_{ij}(t) = \sqrt{Gh^2(i, j) + Gv^2(i, j)} \quad (20)$$

其中,

$$Gh_{ij}(t) = m_{2h}(i, j) C e_h(i, j) w_h^{-1} \quad (21)$$

$$Gv_{ij}(t) = m_{2v}(i, j) C e_v(i, j) w_v^{-1} \quad (22)$$

其中,  $C e_h(i, j)$  和  $C e_v(i, j)$  由文献<sup>[12]</sup> 的 Stimulation 层设计确定。

(5) LPTC: 借助 Stimulation 层中节点在水平、竖直方向的方向量, 获得 LPTC 的输出, 即:

$$\rho(t) = \frac{1}{M+N} \sum_{i=1}^M \sum_{j=1}^N \text{Sigmoid}(Gh(i, j) + Gv(i, j)) \quad (23)$$

需要指出的是, HVNN 与 AFVNN 及 LVNN 在设计上存在较大差异, 主要体现在 4 个方面, 分述如下。

(1) 结构设计不同: HVNN 是在 AFVNN 基础上, 经由利用 LVNN 的 Inhibition、Stimulation 层分别取代 AFVNN 中侧抑制层及  $m_2$  节点层而获得。HVNN 的各层均由  $M \times N$  个神经节点按矩阵形式排列表示, 而 AFVNN 中各层的神经节点数随层数的递增而逐渐减少; LVNN 不含视觉信息投影层, 信息处理层过于单一。

(2) 视觉信息处理存在差异: HVNN 利用 LVNN 的 Inhibition 层执行侧抑制, 而 AFVNN 通过分流抑制动力学模型计算兴奋与抑制量; 也利用 LVNN 的 Stimulation 层生成神经节点的局部方向行为量。

(3) HVNN 利用扰动策略激活状态不变的节点, 而 AFVNN 未含状态扰动策略。

(4) 应用对象不同: AFVNN 和 LVNN 用于运动目标检测, 而 HVNN 适用于求解最优化问题。

## 2.2 状态更新

经由式(16) 获知, 状态矩阵  $\mathbf{A}^{(t)}$  经由 HVNN 中 Retina 层作用后变为  $\mathbf{B}^{(t)}$ 。此状态矩阵中每个状态  $\mathbf{y}_{ij}^{(t)}$  依据 HVNN 的 Stimulation 层中节点  $(i, j)$  输出的行为量  $\lambda_{ij}(t)$  (局部学习率)、LPTC 输出的行为量  $\rho(t)$  (全局学习率), 以及哈里斯鹰优化算法<sup>[20]</sup> 中哈里斯鹰的位置更新策略进行状态更新; 在此, 将哈里斯鹰  $(i, j)$  的位置视为  $\mathbf{A}^{(t)}$  中位置  $(i, j)$  处的状态。  $\mathbf{B}^{(t)}$  中状态  $\mathbf{y}_{ij}^{(t)}$  借助哈里斯鹰的局部与全局捕食策略, 经由式(24) 进行更新:

$$\mathbf{x}_{ij}^{(t+1)} = \begin{cases} \bar{\mathbf{y}}_{ij}^{(t)} & E \geq 1 \\ \mathbf{z}_{ij}^{(t)} & \text{else} \end{cases} \quad (24)$$

其中,  $E$  为最大时间步  $T$  下猎物的逃逸能量, 即:

$$E = 2(2r_1 - 1) \left(1 - \frac{t}{T}\right) \quad (25)$$

在此,  $r_1 \sim U(-1, 1)$ 。  $\bar{\mathbf{y}}_{ij}^{(t)}$  是经由哈里斯鹰  $(i, j)$  在全局学习率  $\rho(t)$  引导下, 经由全局开采策略生成的位置, 即

$$\bar{\mathbf{y}}_{ij}^{(t)} = \begin{cases} \mathbf{x}_{rand}^{(t)} - \rho(t) |\mathbf{x}_{rand}^{(t)} - 2r_1 \mathbf{y}_{ij}^{(t)}| & \zeta \geq 0.5 \\ (\mathbf{y}_{pb}^{(t)} - \mathbf{x}_m^{(t)}) - r_2 [\mathbf{a} + r_3 (\mathbf{b} - \mathbf{a})] & \text{else} \end{cases} \quad (26)$$

其中,  $\zeta, r_2$  和  $r_3$  为  $[0, 1]$  上的随机数,  $\mathbf{a}$  及  $\mathbf{b}$  分别为第 1.3 节所述候选解所在区域  $D$  中各维度的左、右端点值构成的向量;  $\mathbf{x}_{rand}^{(t)}$  是随机生成的状态;  $\mathbf{x}_m^{(t)}$  为  $\mathbf{A}^{(t)}$  中所有状态的平均 (即  $M \times N$  只哈里斯鹰的平均位置);  $\mathbf{y}_{pb}^{(t)}$  为  $\mathbf{B}^{(t)}$  中最好的状态。

在式(26) 中,  $\rho(t)$  作为全局学习率被引入到第一式中, 其主要作用在于调节所有哈里斯鹰的位置转移幅度, 增强全局开采能力; 第二式的目的是强化哈里斯鹰位置的普遍性, 扩大捕食范围。

$\mathbf{z}_{ij}^{(t)}$  是哈里斯鹰  $(i, j)$  在局部学习率  $\lambda_{ij}(t)$  引导下, 经由局部开采策略获得的移动位置, 即:

$$\mathbf{z}^{(t)} = \begin{cases} \mathbf{x}_{pb}^{(t)} - \mathbf{y}_{ij}^{(t)} - \lambda_{ij}(t) \times E |2(1 - r_5) \mathbf{x}_{pb}^{(t)} - \mathbf{y}_{ij}^{(t)}| & |E| \geq 0.5, r \geq 0.5 \\ \mathbf{x}_{pb}^{(t)} - \lambda_{ij}(t) \times E |\mathbf{x}_{pb}^{(t)} - \mathbf{y}_{ij}^{(t)}| & |E| < 0.5, r \geq 0.5 \\ \min(\mathbf{y}_{ij}^{(t)}, \mathbf{Y}_{ij}^{(t)}, \mathbf{Z}_{ij}^{(t)}) & \text{else} \end{cases} \quad (27)$$

其中,  $r \sim U(0,1)$ , 进而推得:

$$\mathbf{Y}_{ij}^{(t)} = \mathbf{x}_{pb}^{(t)} - \lambda_{ij}(t) \times E | 2(1 - r_s) \mathbf{x}_{pb}^{(t)} - \mathbf{y}_{ij}^{(t)} | \quad (28)$$

$$\mathbf{Z}^{(t)} = \mathbf{Y}_{ij}^{(t)} + 0.01 \times \frac{u\sigma}{|v|^{\beta}} \mathbf{I} \quad (29)$$

$$\sigma = \left\{ \frac{\Gamma(1 + \beta) \sin(\pi\beta/2)}{\Gamma[(1 + \beta)/2] \beta \times 2^{(\beta-1)/2}} \right\}^{1/\beta} \quad (30)$$

其中,  $\mathbf{I}$  是由  $0 \sim 1$  之间随机生成的  $p$  个值构成的列向量;  $u$  和  $v$  为  $[0,1]$  上的随机数;  $\beta$  为常数, 设置为 1.5;  $\sigma$  是 Levy 飞行函数;  $\min(x,y)$  表示 2 个  $p$  维向量  $\mathbf{x}$  与  $\mathbf{y}$  的相同位置处分量的最小值构成的向量。

在式(27)中, 第一式经由局部学习率  $\lambda_{ij}(t)$  被引入到包围猎物搜索策略而获得, 其目的在于借助  $\lambda_{ij}(t)$  引导劣质哈里斯鹰 ( $i, j$ ) 逐渐朝更有利于捕食的方向转移; 第二式经由搜索猎物策略而获得, 其目的是增强状态转移的多样性; 第三式通过引导哈里斯鹰进行位置信息交互, 确保种群位置的多样性。

### 2.3 HVENN 的算法描述

HVENN 作为一种循环神经网络, 由 HVNN 和一个  $M \times N$  神经节点矩阵顺次连接而成。HVNN 以第  $t-1$ 、 $t$  时刻的状态矩阵  $\mathbf{A}^{(t)}$  及  $\mathbf{A}^{(t-1)}$  对应的灰度图的差图作为输入, 依次经由 Retina、Cartridge、Inhibition、Medulla、Stimulation 及神经元 LPTC 做视觉信息处理, 获得的局部学习率和全局学习率在以上状态转移策略下, 引导状态转移。算法描述如下。

**步骤 1** 参数设置: 灰度图的分辨率  $M \times N$ , 最大时间步  $T$ 。

**步骤 2** 置  $t \leftarrow 1$ , 随机生成  $M \times N$  个候选解, 构成初始状态矩阵  $\mathbf{A}^{(t)}$ 。

**步骤 3** 计算  $f(\mathbf{A}^{(t)})$ , 且  $\mathbf{A}^{(t)}$  中灰度值或目标值最小的状态, 记  $\mathbf{x}_{pb}^{(t)}$  和  $\mathbf{x}_{gb}$ 。

**步骤 4** 执行 HVNN, 将  $\mathbf{A}^{(t)}$  转移为  $\mathbf{B}^{(t)}$ , 同时获得局部学习率  $\lambda_{ij}(t)$  和全局学习率  $\rho(t)$ 。

**步骤 5**  $\mathbf{A}^{(t)}$  中所有状态依据式(24) 执行状态更新, 获得状态矩阵  $\mathbf{A}^{(t+1)}$ 。

**步骤 6** 计算  $\mathbf{A}^{(t+1)}$  对应的灰度图  $f(\mathbf{A}^{(t+1)})$ , 并借助  $\mathbf{A}^{(t+1)}$  中的状态, 更新  $\mathbf{x}_{gb}$ 。

**步骤 7** 置  $t \leftarrow t + 1$ , 若  $t \leq T$ , 则返回步骤 4; 否则, 结束且输出  $\mathbf{x}_{gb}$ 。

以上算法利用 HVNN 输出的全局学习率控制状态的变化幅度, 防止因状态转移的幅度过大而导致状态远离最优解的位置, 增强算法的收敛性。同时, 对状态分量进行随机扰动, 防止算法过早陷入局

部搜索, 增强算法的全局搜索能力。另外, 局部学习率被用于调节状态局部转移的幅度, 增强算法的局部搜索能力。

HVENN 的计算复杂度由 HVNN 和状态更新策略确定。在 HVNN 中, Retina 层共执行  $MN$  次加减运算; Cartridge 层共运行  $28MN$  次; Inhibition 层共执行  $18MN$  次加减法; Medulla 层执行  $27MN$  次运算; Stimulation 层共执行  $12MN$  次乘法运算、 $10MN$  次加减运算、 $MN$  次求最大值和取绝对值运算; Lobula 层共执行  $7MN$  次运算。状态更新模块需要执行  $15MNp$  次运算以及  $MNm_p$  次目标函数评价, 在此  $m_p$  表示优化问题中目标函数的计算次数。于是, 在一个执行周期内, HVENN 的运行次数为  $MN(m_p + 15p + 104)$ 。因此, HVENN 的计算复杂度由  $M, N, p$  和  $m_p$  确定。又因要求  $M, N$  的取值较小, 所以算法的效率由优化问题本身而定。

## 3 数值实验

在配置为 Windows10/CPU 3.7 GHz/RAM 4.0 GB/Matlab R2018a 环境下展开数值实验。首先, 借助基准函数和参与比较的算法测试 HVENN 求解函数优化问题的性能; 其次, 将 HVENN 应用于 DV-Hop 节点定位问题, 测试其应用效果和所获锚节点与未知节点之间的距离计算模型设计的合理性。

### 3.1 HVENN 的性能测试

选取 6 种元启发式随机搜索算法参与 HVENN 的比较, 即灰狼优化 (GWO)<sup>[20]</sup>、麻雀搜索算法 (SSA)<sup>[9]</sup>、飞蛾扑火优化 (MFO)<sup>[21]</sup>、多元宇宙优化 (MVO)<sup>[22]</sup>、蝴蝶优化算法 (BOA)<sup>[23]</sup> 及哈里斯鹰优化 (HHO)<sup>[24]</sup>。测试事例包括维度均为 30 的 13 种基准测试函数优化问题  $f_1 \sim f_{13}$ <sup>[22]</sup>, 其中  $f_1 \sim f_7$  是单峰值函数,  $f_8 \sim f_{13}$  是含大量局部最优解的多峰值函数。参与比较算法的种群规模均为 100, 其它参数设置源于相应文献。HVENN 的参数设置为  $M = N = 10$ 。对于每个测试问题, 各算法的终止条件是最大迭代数为 500 或获得的最好解的目标值为 0, 且均独立运行 30 次。各算法获得每个问题的 30 个最好目标值的均值、均方差及在显著性水平为 5% 下的  $t$  检验值用于算法的比较分析, 见表 1; 以  $f_1, f_3, f_5, f_7, f_9, f_{11}$  为例, 算法的搜索曲线如图 3 所示。

经由表 1 的均值和方差可知, HHO 和 HVENN 求解  $f_1 \sim f_{13}$  所获解的质量明显优于其它参与比较的算法所获解质量, 且均能稳定地获得优化问题的最优解或近似解, 搜索效果稳定。相对而言,

HVENN 比 HHO 获得的解的精度更高,且获得最优解的基准测试问题个数更多,尤其是求解较为困难的  $F_8$ , 其搜索效果较为稳定。其次,GWO 和 BOA 求解以上 13 个优化问题时,无论是解的质量、还是搜索效果的稳定性,均优于 SSA、MFO 和 MVO,分析

其原因在于二者仅分别不能有效求解 2、4 个问题。最后, MFO 不能有效求解 13 个测试问题; SSA 和 MVO 不能求解 9 个测试问题。结合表 1 中  $t$  检验值可知, HVENN 求解以上问题的性能最好, HHO 次之, 而 MFO 的性能最差。

表 1 各算法求解每个问题时,独立运行 30 次后获得的统计结果比较

Tab. 1 Comparison of the statistical results obtained after 30 independent runs when each algorithm solves each problem

$F$	$Q$	GWO	SSA	MFO	MVO	BOA	HHO	HVENN
$F_1$	$\mu$	4.97E-41	1.78E-07	3.68E+03	5.74E-01	1.26E-11	2.11E-99	<b>0.00E+00</b>
	$\sigma$	5.50E-41	1.84E-07	6.69E+03	1.51E-01	7.92E-13	9.11E-99	0.00E+00
$F_2$	$\mu$	4.78E-24	1.58E+00	3.15E+01	4.39E+00	1.23E-11	8.43E-51	<b>1.30E-195</b>
	$\sigma$	3.57E-24	1.40E+00	2.12E+01	2.07E+01	1.29E-09	2.89E-50	0.00E+00
$F_3$	$\mu$	1.23E-11	1.95E+03	2.15E+04	6.43E+01	1.23E-11	7.94E-74	<b>0.00E+00</b>
	$\sigma$	5.32E-11	1.34E+03	1.27E+04	2.22E+01	1.18E-12	4.31E-73	0.00E+00
$F_4$	$\mu$	1.88E-10	1.05E+01	6.98E+01	1.17E+00	6.14E-09	1.55E-49	<b>4.25E-178</b>
	$\sigma$	2.03E-10	3.14E+00	9.05E+00	3.57E-01	3.46E-10	8.05E-49	0.00E+00
$F_5$	$\mu$	2.66E+01	4.93E+02	8.01E+06	3.53E+02	2.88E+02	1.37E-02	<b>6.69E-03</b>
	$\sigma$	8.70E-01	1.09E+03	2.44E+07	6.10E+02	2.49E-02	2.62E-02	8.43E-03
$F_6$	$\mu$	2.27E-01	1.53E-07	1.02E+03	6.06E-01	1.08E+00	2.63E-04	<b>4.84E-05</b>
	$\sigma$	2.55E-01	2.05E-07	3.08E+03	1.74E-01	0.38E+00	4.72E-04	1.04E-04
$F_7$	$\mu$	5.69E-04	1.86E-01	1.28E+00	2.06E-02	1.20E-02	1.55E-04	<b>1.40E-04</b>
	$\sigma$	3.76E-04	8.25E-02	1.03E+01	6.97E-03	5.79E-04	1.56E-04	1.26E-04
$F_8$	$\mu$	-6.25E+03	-7.56E+03	-8.49E+03	-7.87E+03	-2.10E+03	-1.25E+04	<b>-1.26E+04</b>
	$\sigma$	9.31E+02	7.55E+02	8.70E+02	5.21E+02	2.26E+02	3.18E+02	2.27E+00
$F_9$	$\mu$	8.60E-01	5.33E+01	1.66E+02	1.20E+02	3.24E+01	<b>0.00E+00</b>	<b>0.00E+00</b>
	$\sigma$	3.04E+00	2.06E+01	2.77E+01	3.17E+01	1.79E+01	0.00E+00	0.00E+00
$F_{10}$	$\mu$	2.75E-14	2.49E+00	1.39E+01	1.21E+00	2.75E-09	8.88E-16	8.88E-16
	$\sigma$	4.35E-15	7.87E-01	7.04E+01	5.76E-01	1.28E-09	<b>0.00E+00</b>	<b>0.00E+00</b>
$F_{11}$	$\mu$	2.31E-03	1.86E-02	2.19E+01	6.87E-01	1.98E-13	<b>0.00E+00</b>	<b>0.00E+00</b>
	$\sigma$	4.92E-03	1.26E-02	3.86E+01	8.78E-02	1.75E-13	0.00E+00	0.00E+00
$F_{12}$	$\mu$	1.37E-02	6.91E+00	1.04E+01	1.47E+00	1.21E-01	<b>5.26E-06</b>	5.84E-06
	$\sigma$	1.04E-02	3.75E+00	7.06E+00	1.31E+00	5.24E-02	5.89E-06	1.04E-05
$F_{13}$	$\mu$	2.10E-01	1.58E+01	1.37E+07	8.72E-02	4.11E-01	1.03E-04	<b>3.29E-05</b>
	$\sigma$	1.53E-01	1.59E+01	7.49E+07	3.68E-02	1.65E-01	1.92E-04	6.33E-05
$w/t/l$		13/0/0	13/0/0	13/0/0	13/0/0	13/0/0	9/2/2	/

注:  $w, t, l$  分别表示 HVENN 显著优于、同等性能于、劣于被比较算法的问题数,  $\mu$  和  $\sigma$  分别表示各算法求解一个测试问题获得的目标值的均值和均方差



图3中,  $n$ 表示迭代次数,  $f$ 表示适应度值。图3表明,相较于参与比较的算法, HVENN的全局搜索能力强且收敛速度最快; HHO相比于其它参与比较的算法,搜索速度快,但获得的解的精度有待提

高,且易于收敛到近似解; BOA的进化能力比GWO、SSA、MFO、MVO的能力强,但收敛速度比GWO的慢。

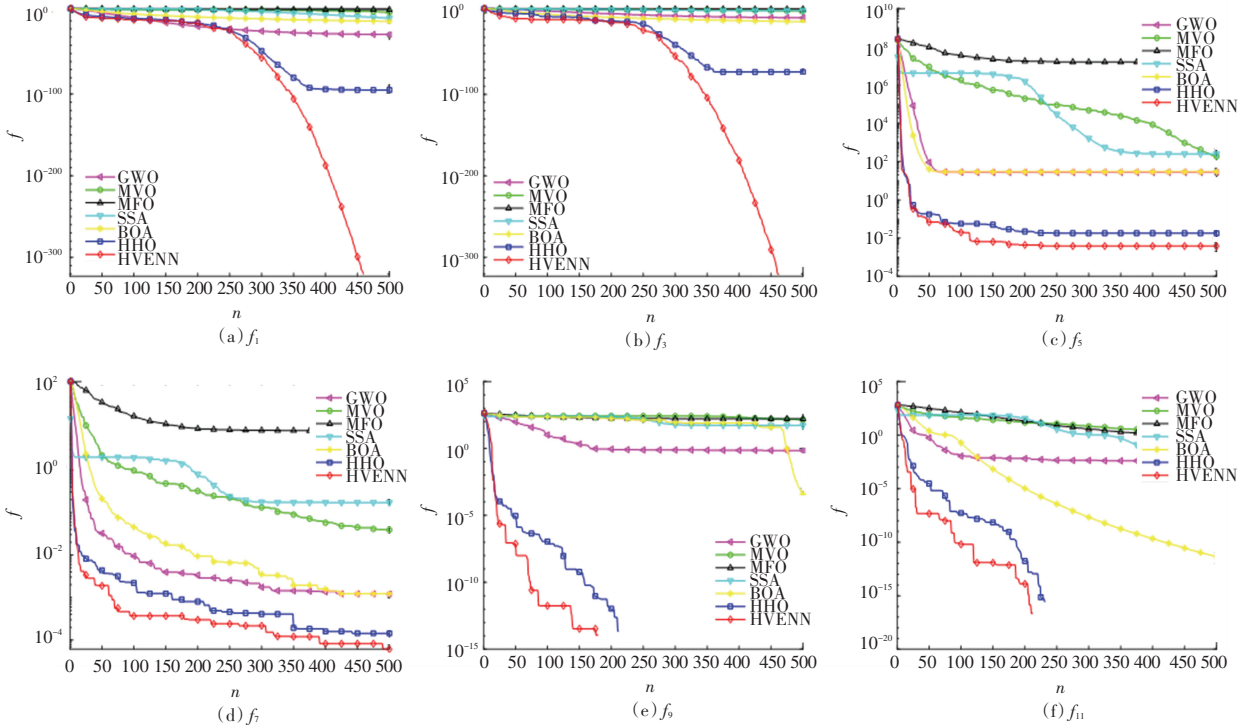


图3 算法的平均搜索曲线比较

Fig. 3 Comparison of the average search curves of the algorithms

### 3.2 DV-Hop 定位

为测试 HVENN 求解 1.2 节中 DV-Hop-M 的性能,选取经典 DV-Hop 算法<sup>[4]</sup>、MVODV-hop<sup>[21]</sup>、GWODV-Hop<sup>[9]</sup>、BOADV-Hop<sup>[23]</sup>参与节点定位误差比较。设置锚节点和未知节点所在矩形的面积为  $100\text{ m} \times 100\text{ m}$ ,且随机抛撒  $m$  个锚节点及  $n$  个未知节点,如图4所示。图4中,“\*”表示红色信标节点,“o”表示黑色未知节点。依据以下节点定位的平均误差指标检测各算法的性能:

$$Err = \frac{\sum_{i=1}^n \sqrt{(x_i - \bar{x}_i)^2 + (y_i - \bar{y}_i)^2}}{n \times R} \times 100\% \quad (31)$$

其中,  $(x_i, y_i)$  和  $(\bar{x}_i, \bar{y}_i)$  分别是未知节点  $i$  的估计和实际位置;  $R$  是通信半径;  $Err$  是  $n$  个未知节点的平均定位误差。式(31)表明,  $Err$  越小,则算法的节点定位精度越高;反之,精度越低。这里,给出研究内容详述如下。

(1)情形1:锚节点数量。设置  $R = 25\text{ m}$ ,锚节点数为  $m = 10 + 5k$  ( $0 \leq k \leq 8$ ),未知节点数为  $n = 100 - m$ ,节点总数为100。

在给定的  $k$  值下,各算法求解 DV-Hop-M 100次,获得的100个  $Err$  的均值形成的曲线如图5(a)所示。经由图5(a)获知,随着  $k$  的增大,锚节点数量逐渐增多,因而未知节点数逐渐减少,定位精度逐渐提高。HVENN的  $Err$  为28.7%,参与比较的4种算法的  $Err$  分别为66.5%、57.2%、52.2%及45.3%。因此,HVENN的节点定位偏差最小,而DV-Hop的节点定位偏差最大。

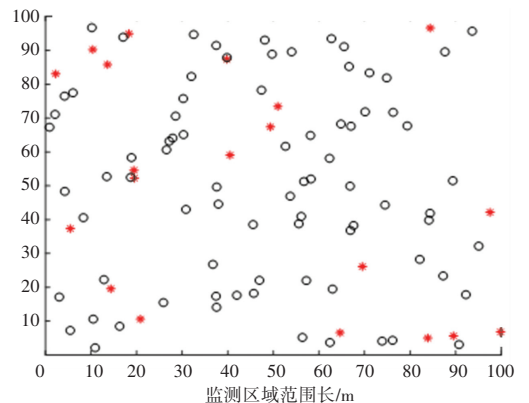
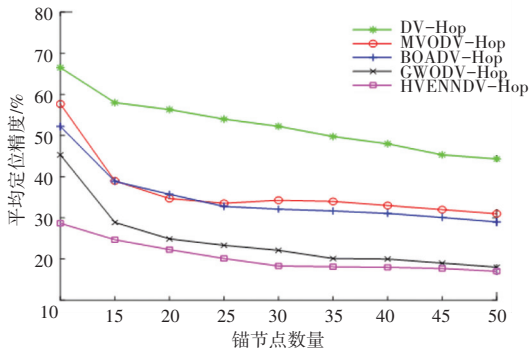


图4 网络节点分布:  $m=20, n=80$

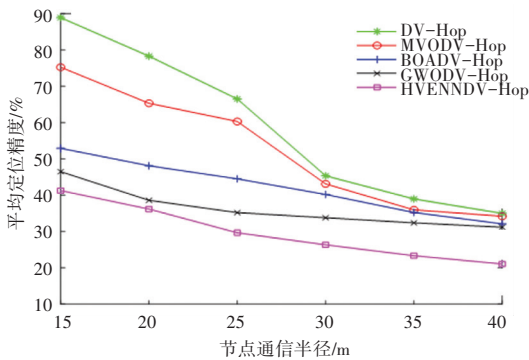
Fig. 4 Network node distribution:  $m=20, n=80$



(2) 情形 2: 节点通信半径。矩形区域面积同上; 设置  $m = 30, n = 70$ , 通信半径  $R = 15 + 5k$  (单位为 m),  $0 \leq k \leq 5$ 。同上, 在给定  $k$  值下, 各算法获得的定位精度曲线如图 5(b) 所示。经由图 5(b) 获知, 当  $20 \leq R \leq 35$  时, 随着通信半径的加大, 未知节点收集到的定位信息越多, 网络连通性越强, 定位精度逐渐提高; 当  $R > 35$  时, 各算法获得的节点定位精度逐渐趋于平稳。主要原因在于, 当通信半径较小时, 每个节点的相邻节点较少, 网络连通性较低, 导致平均跳距存在一定误差, 进而定位效果较差。当通信半径增大时, 节点间的跳距变小, 从而定位误差减小; 当通信半径达到 35 m 时, 区域内的 2 个节点间的跳距都在 3 跳之内, 所以随着  $R$  的增大, 定位误差趋于稳定且较小。相较于其它基准对比算法, HVENN 的定位精度最高, 但 DV-Hop 的定位精度最低。



(a) Err 均值曲线



(b) 定位精度曲线

图 5 定位误差对比

Fig. 5 Comparison of positioning errors

### 4 结束语

为了解决 DV-Hop 定位模型存在定位误差大的问题, 在改进节点跳数和平均跳距计算模型基础上, 将未知节点的定位问题描述为非约束确定性优化问题, 进而利用果蝇和蝗虫视觉系统的信息处理机制获得混合视觉神经网络, 并将其输出与哈里斯

鹰的位置更新策略结合, 获得混合视觉进化神经网络(HVENN)。理论分析表明, 该神经网络的计算复杂度由其分辨率和优化问题的维度确定。实验结果表明, 将视觉神经网络与元启发方法结合, 探讨求解优化问题的视觉进化神经网络, 是一条切实可行的新路径; 相比于其它基准对比算法, HVENN 在求解精度和收敛速度方面均有明显优势, 且能较好地解决 WSN 的节点定位问题, HHO 次之, 而其它基准对比算法的性能较差。另外, 本文仅涉及到静态环境下的节点定位问题及其求解, 至于如何探讨不确定环境下的节点定位以及 HVENN 是否具有适应环境的能力, 仍需做后续进一步的研究探讨。

### 参考文献

- [1] MAO Guoqiang, FIDAN B, ANDERSON B. Wireless sensor network localization techniques [J]. Computer Networks, 2007, 51(10): 2529-2553.
- [2] ZHU Aijun, CHEN Duanyong, XU Chuanpei, et al. A fault check graph approach for photonic router in network on chip [C]// 2018 IEEE 27th Asian Test Symposium (ATS). IEEE, 2018: 13-18.
- [3] ZHAO Wei, SU Shoubao, SHAO Fei. Improved DV-Hop algorithm using locally weighted linear regression in anisotropic wireless sensor networks [J]. Wireless Personal Communications, 2018, 98(4): 3335-3353.
- [4] NICULESCU D, NATH B. DV based positioning in Ad Hoc networks [J]. Telecommunication Systems, 2003, 22(1/2/3/4): 267-280.
- [5] KAUR A, KUMAR P, GUPTA G P. A novel DV-Hop algorithm based on Gauss-Newton method [C]// International Conference on Parallel, Distributed and Grid Computing. Wagnaghat, India: IEEE, 2016, 25(2): 625-629.
- [6] 张斌, 史文进, 李冠迪, 等. 无线传感网络中基于 RSSI 质心定位的改进算法 [J]. 重庆理工大学学报(自然科学), 2017, 31(03): 132-136.
- [7] MA Shuli, ZHAO Jianping. Multi communication ranges DV-Hop localization algorithm for wireless sensor network [J]. Chinese Journal of Sensors and Actuators, 2016, 29(4): 593-600.
- [8] 褚银菲, 吕惠芳. 基于测距修正的差分进化的 DV-Hop 定位算法 [J]. 国外电子测量技术, 2021, 40(06): 33-37.
- [9] 印雷, 顾德, 刘飞. 基于改进麻雀搜索算法优化的 DV-Hop 定位算法 [J]. 传感技术学报, 2021, 34(05): 670-675.
- [10] FU Qinqing, HU Cheng, PENG Jigen, et al. Shaping the collision selectivity in a looming sensitive neuron model with parallel on and off pathways and spike frequency adaptation [J]. Neural Networks, 2018, 106(1): 127-143.
- [11] MISSLER J M, KAMANGER F A. A neural network for pursuit tracking inspired by the fly visual system [J]. Neural Networks, 1995, 8(3): 463-480.
- [12] 郭小沛. 基于蝗虫视觉神经的微型机器人避障方法研究 [D]. 武汉: 武汉科技大学, 2021: 28-32.
- [13] 胡滨. 基于蝗虫视觉神经的运动模式识别研究及应用 [D]. 贵阳: 贵州大学, 2019: 25-38.