

融合神经网络及麻雀算法的机器人避障研究

朱金坛

(西安铁路职业技术学院 电子信息学院, 西安 710014)

摘要: 针对麻雀搜索算法 (SSA) 在机器人避障研究中, 存在提前收敛于局部最优难以跳出、初始种群分布不够广泛、平衡能力差等问题对其进行改进; 通过三层神经网络对规划环境进行栅格化建模; 引入 Halton 序列得到初代种群分布, 得到分布更广、更遍历的个体位置, 提升后期寻优速度和效率; 使用布朗运动优化麻雀位置更新的步长调节, 帮助算法脱离局部优解, 同时平衡全局切换局部的搜索节奏; 利用 clothoid 曲线法平滑路径, 得到满足机器人机械性能的路径; 经 6 个标准函数验证和 Wilcoxon 检验 P 值对比可知, 改进后的算法相较于 SSA 和 CSSA 算法各项指标得到明显优化, 且具有和 SSA 同一水平的复杂度。通过地图仿真得到平滑后的机器人避障路径。

关键词: 麻雀搜索算法; 低差异序列; 神经网络; 布朗运动

Research on Robot Obstacle Avoidance Based on Neural Network and Sparrow Algorithm

ZHU Jintan

(School of Electronic Information, Xi'an Railway Vocational and Technical Institute, Xi'an 710014, China)

Abstract: Aiming at the problems of a sparrow search algorithm (SSA) in a robot obstacle avoidance research, such as early convergence to the local optimal difficult to jump out, insufficient distribution of the initial population, poor balance ability, in order to improve the questions, a three-layer neural network was used to model the planning environment raster. The Halton sequence was introduced to obtain the initial population distribution, obtain the individual locations with wider distribution and more ergodic distribution, and improve the speed and efficiency of later optimization. The Brownian motion was used to optimize the step size adjustment of the sparrow position update, which helped the algorithm out of local optimal solution and balanced the local search rhythm of global switch. The method of the clothoid curve was used to smooth the path, and obtain the satisfied path with the mechanical properties of the robot. Compared with the algorithms of the SSA and chaos sparrow search algorithm (CSSA), the improved algorithm is significantly optimized in the P-values of the Wilcoxon test and six standard functions, and has the same level of time complexity as the SSA. The smooth obstacle avoidance path of the robot is obtained by the map simulation.

Keywords: SSA; low difference sequence; neural network; brownian motion

0 引言

机器人自主运行的核心技术就是能避开障碍, 具体来说是指机器人在无人干预的情况下, 自主实现计划中要求的各点间可行路径的搜索与运行。规划过程需要结合自身运行动力特性和环境约束, 根据自身能耗和速度特征, 考虑环境威胁等条件, 自主规划出最优路径。其中, 路径的平滑度、行程长短、机器人能量消耗则是对该条路径优劣的评估标准。从规划环境尺度角度可分为全局避障和局部避障两大类。局部路径避障通过传感器获取环境数据, 选择一条当下节点至下一节点的子路径, 常用方法有滚动窗口法^[1]、Dijkstra 算法^[2]、人工势场法^[3]等。全局避障指在

掌握运动环境的前提下, 利用算法自主生成优化路径, 如基于栅格图对环境建模的方法^[4]、利用可视图法模拟实际环境的方式^[5]以及基于 open 表的 A* 算法^[6]等。

上述传统算法虽然容易实现, 但存在实时性差、适用范围小、复杂环境计算量过大的问题。近年群智能算法的研究发展为解决机器人避障问题提供了更多解决方案。如: 游达章^[7]等人融合粒子群与灰狼算法, 引入量子协同改善了算法在处理路径规划问题的精度; 周晟^[8]等人引入随机分形自适应搜索策略优化了仓储机器人多障碍多目标环境的路径规划问题; 潘纹^[9]基于动态步长果蝇算法研究了自动导引车的路径规划问题。还有其他如天牛须算法^[10]、鲸鱼优化算法^[11]等仿生群体算法也都开始用于路径规划与

收稿日期: 2022-12-12; 修回日期: 2023-01-18。

基金项目: 2022 年陕西省地方课程教材及教辅资源研究课题 (20220200800)。

作者简介: 朱金坛 (1981-), 男, 陕西渭南人, 硕士, 副教授, 主要从事大数据及人工智能方向的研究。

引用格式: 朱金坛. 融合神经网络及麻雀算法的机器人避障研究[J]. 计算机测量与控制, 2023, 31(4): 258-263, 288.

避障研究当中。2020 年薛建凯等^[12]首次提出麻雀搜索算法，算法主要以数学公式模拟麻雀群体觅食过程，具有速度快、精度高的特点。但由于收敛速度快也更容易提早收敛于局部极值点而无法跳出、初始种群分布不够广泛以及平衡能力差等缺点。该算法在 CT 图检测^[13]、电池参数优化^[14]、算法参数优化^[15]和成像偏移预测^[16]等实际问题。但 SSA 仍存在过度依赖初始种群，收敛前种群多样性不足，易陷入局部极值且无法跳出等不足。

为解决以上问题，许多相关研究针对不同角度进行了优化改进。如 G. Liu 等人^[17]引入混沌策略增加多样性，使用权重和突变策略加快寻优。欧阳城添等人^[18]融合折射学习，以此初始化种群，丰富多样性，同时引入疯狂算子使得算法寻优能力得以提升。这些算法虽也有一些算法上的提升，但并没有从根本优化寻优机制，而是把结果交给不确定性较大的混沌随机机制，也没有考虑局部与全局的平衡，需要以时间复杂度为代价去优化算法性能。

因此针对算法提早收敛于局部、初始种群分布不够广泛和平衡能力差等问题，提出一种融合神经网络的改进麻雀搜索算法。首先，通过神经网络对规划环境进行栅格化建模，其次引入低差异的 Halton 序列进行麻雀种群初始化，可使得初代种群分布广泛且均匀，有很好的遍历性；再次，采用布朗运动步长策略优化传统 SSA 算法平衡能力差的缺陷，实现前期全局过程搜索范围广，中后期搜索精度高的平衡与切换，同时可通过布朗运动步长跳出局部收敛；最后，为了避免机器人实际运行过程出现急转急停问题，引入 clothoid 曲线法进行最终的路径平滑，避免了尖锐拐弯，减少实际运行时轮轴磨损和能耗。将综合改进后的算法与原始算法进行时间复杂度对比，并使用 6 种函数对几种常见算法进行横向对比，最后将改进后算法与 SSA 算法和 CSSA 算法^[19]进行不同尺度的环境规划对比，综合评价算法的机器人避障效果指标性能。

1 麻雀搜索算法

麻雀搜索算法 (SSA) 是一种启发式算法，由麻雀搜索食物过程中的行为模式启发而来，有算法稳定、寻优速度较快的特点。不仅优于传统粒子群算法，对于新型的算法如灰狼算法、引力搜索算法也具有一定优势。SSA 真个种群共有 3 种角色，分别是发现者、尾随者和警戒者，三者有不同的任务分工，发现者负责探索食物方位，并将食物位置共享给跟随者，跟随者的任务则是尾随发现者，当收到发现者的位置更新后，则前往新位置准备觅食，同时，为了保证种群安全，种群中部分个体会在觅食前进行警戒，当收到危险信号时则会通知种群放弃觅食。

对于算法而言，发现者即执行的全局探测任务，跟随者即对更新的位置信息进行局部寻优，当陷入局部极小值时，有警戒者负责释放危险信号跳出局部，重新进行全局寻优。具有预警机制也是该算法优于其他算法的核心，使得 SSA 相较于其他算法更不易陷入局部最优解，更容易取

得全局最优解。

SSA 算法中的发现者、追随者以及警戒者，分别按照各自规则进行位置更新，更新规则如式 (1) ~ (3) 所示：

$$X_i^{t+1} = \begin{cases} X_i^t \times \exp\left(\frac{-i}{\xi \times iter_{max}}\right), & R_2 < ST \\ X_i^t + q \times L, & R_2 \geq ST \end{cases} \quad (1)$$

其中： t 为当前代数； X_i^{t+1} 表示在 $t+1$ 代第 i 只麻雀的适应值； $iter_{max}$ 是最大迭代数； $\xi \in (0, 1)$ 是一个随机数； R_2 表示警戒值； ST 表示安全阈值； q 是一个服从正态分布的随机数； L 是一个一行多维的全一矩阵。

$$X_i^{t+1} = \begin{cases} q \times \exp\left(\frac{X_{worst}^t - X_i^t}{i^2}\right), & i > \frac{n}{2} \\ X_i^t + |X_i^t - X_p^t| \times A^+ \times L, & i \leq \frac{n}{2} \end{cases} \quad (2)$$

其中： X_i^t 表示被发现者占据的最佳位置； X_{worst}^t 表示当前最差位置； A 是一个各元素为 1 或 -1 的一行多维矩阵。

$$X_i^{t+1} = \begin{cases} X_{best}^t + \beta \times |X_i^t - X_{best}^t|, & f_i > f_g \\ X_i^t + K \times \left(\frac{|X_i^t - X_{worst}^t|}{(f_i - f_w) + \epsilon}\right), & f_i \leq f_g \end{cases} \quad (3)$$

其中： X_{best}^t 是当前最优位置； β 是步长参数； $K \in (0, 1)$ 是一个随机数； f_i 是当前麻雀的适应度， f_g 和 f_w 是当前最优适应度和最差适应度； ϵ 为常数，本文取 10^{-8} 避免分母为零。

2 改进的麻雀搜索算法

2.1 三层神经网络建模

首先有三点假设前提：一是机器人运动范围并非无限，而是在限定范围内自主运行；二是以机器人最大尺度半径膨胀障碍物，得到栅格环境；三是将机器人视为无尺度无边界的质点处理。三层神经网络如图 1 所示，数学表达如式 (4) 所示：

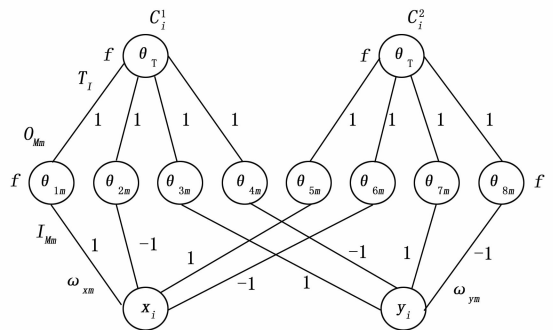


图 1 神经网络的环境建模

$$\left. \begin{aligned} C_k^i &= f(T_l) (k = 1, 2) \\ T_l &= \sum_{m=1}^M O_{Mm} + \theta_T \\ O_{Mm} &= f(I_{Mm}) \\ I_{Mm} &= \omega_{xm} x_i + \omega_{ym} y_i + \theta_{Mm} \end{aligned} \right\} \quad (4)$$

式中，最后一层网络的输入表示为 T_l ，输出表示为 $C_k^i (k = 1, 2)$

2), θ_r 代表当前节点的临界值, 一般为负数, 该模型取 $1/2$ 减去障碍物边数。 I_{M_m} 和 O_{M_m} 是中间层结点的 IO 口, ω_{xm} 和 ω_{ym} 是权重参数, $\theta_m = (i = 1, 2, \dots, 8)$ 中间层节点的临界值。 $f(\cdot)$ 为激发函数, 该模型使用阶跃函数, (x_i, y_i) 为机器人坐标。

其中: $\theta_m = (i = 1, 2, \dots, 8)$ 计算路径如下:

$$\left. \begin{aligned} \theta_{1m} &= -(2 + v_{kx}t_i), \theta_{2m} = 4 + v_{kx}t_i \\ \theta_{3m} &= -(6 + v_{ky}t_i), \theta_{4m} = 8 + v_{ky}t_i \\ \theta_{5m} &= -(7 + v_{kx}t_i), \theta_{6m} = 9 + v_{kx}t_i \\ \theta_{7m} &= -(2 + v_{ky}t_i), \theta_{8m} = 4 + v_{kx}t_i \end{aligned} \right\} \quad (5)$$

式中, 假设机器人的切线方向速度在 $[t_i, t_{i+1}]$ 内不变。 V_{kx} 和 V_{ky} 为速度分量。 t_i 与 t_{i+1} 之间的关系如下式:

$$t_{i+1} = t_i + \frac{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}}{v_{rob}(t_i)} \quad (6)$$

式中, 机器人在 t_i 时速度为 v_{rob} , 且在 $[t_i, t_{i+1}]$ 速度不变。

模型第三层神经结构的输出只有 0 和 1, 当 $C_k^*(x_i, y_i) = 1 (k = 1, 2)$ 时, 则代表障碍物侵限, 反之为安全路径。 避碰检测算法如下所示:

$$Z = \sum_{i=1}^{P_r} \sum_{j=1}^{P_s} C_{R_{ij}} \quad (7)$$

式中, 避碰检测值 Z 为 0 时, 代表可行路径, 取正值时则产生碰撞, 路径不可行。 障碍物数目为 P_R , 路径等分点个数为 P_e 。 此模型可同时计算出当前路径与所有障碍物之间的碰撞结果。

2.2 低差异序列初始化

初代种群分布的广泛性与均匀性直接影响算法收敛的速度。 初代种群分布越广, 全局探索的速度越快, 局部寻优的精度越高。 标准 SSA 中初代麻雀种群位置过于随机, 均匀性过低、许多区域存在盲区, 不够遍历。 因此引入一种基于超均匀分布的均匀分布随机数组 Halton 序列, 以此代替随机数进行种群初始化操作。

Halton 序列主要有如下三步计算过程:

1) 假定 $p \geq 2$ 且为质数, 对任意整数 $n, n \in (1, N)$, 可以 p 为基表达出如式 (8) 的形式:

$$\begin{aligned} n &= \sum_{i=0}^m b_i p^i = b_m p^m + \dots + b_1 p^1 + b_0, \\ b_i &\in \{0, 1, \dots, p-1\} \end{aligned} \quad (8)$$

2) 已知 b_i 与 p , 基本逆序函数如式 (9):

$$\begin{aligned} \varphi_p(n) &= (0. b_0 b_1 \dots b_m)_p = b_0 p^{-1} + \\ & b_1 p^{-2} + \dots + b_m p^{-m-1} \end{aligned} \quad (9)$$

3) d 维空间的 Halton 序列计算如式 (10):

$$H(n) = [\varphi_{p_1}(n), \varphi_{p_2}(n), \dots, \varphi_{p_d}(n)] \quad (10)$$

在解空间上下限中生成一个取值范围在 0 和 1 之间的随机数 K_n , 初始位置 X_n 如式 (11) 所示:

$$X_n = X_{\min} + K_n \cdot (X_{\max} - X_{\min}) \quad (11)$$

如图 2 所示, 可以清楚看出二维空间中 Halton 序列相对于伪随机数法可将种群分布得更加均匀。

由图 2 可看出, Halton 序列初始化种群相比伪随机数

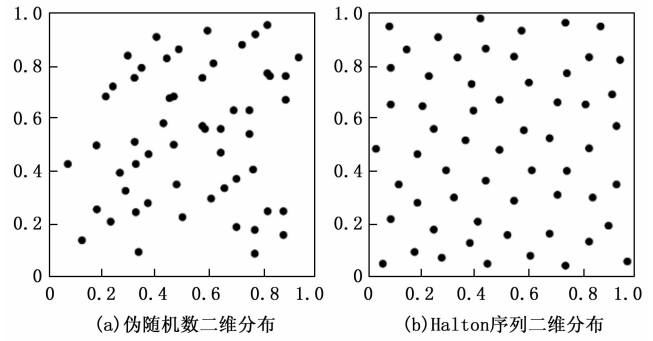


图 2 二维位置分布对比图

序列得到分布更均匀、遍历性更广的初始化麻雀群体。

2.3 布朗运动步长策略

传统 SSA 算法的迭代步长由一个符合均匀分布的随机数作为步长因子, 存在随机性过大的问题, 不能很好的平衡前期全局开发与中后期局部开发的过程切换。 因此引入布朗运动步长策略, 满足前期全局过程搜索范围足够广, 中后期搜索精度足够高的运行要求, 当算法收敛于局部最优值, 也可通过布朗运动策略跳出局部最优, 脱离算法停滞。

布朗运动的步长服从均值为 0 方差为 1 的高斯分布, 其本质是一个均匀可控的步长随机过程, 概率密度函数如式 (12) 所示:

$$f_B(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (12)$$

麻雀位置更新公式为:

$$X_{i,j}^{t+1} = X_{i,j}^t + PR(R_B(X_{best}^t - R_B X_{i,j}^t)) \quad (13)$$

式 (13) 中, 布朗运动步长以 R_B 表示; P 等于 $1/2$; R 为 $(0, 1)$ 间随机数。

2.4 clothoid 曲线法的路径平滑

clothoid 曲线广泛运用于车辆行驶轨迹设计, 路径规划等方向, 是一种曲率半径随弧长线性变化的曲线, 可以在直线段与曲线段平滑过渡。 clothoid 曲线的数学表达如式 (14) 所示:

$$\begin{cases} x = x_0 + \int_0^s \cos(\theta_0 + k_0 \tau + \frac{1}{2} c \tau^2) d\tau \\ y = y_0 + \int_0^s \sin(\theta_0 + k_0 \tau + \frac{1}{2} c \tau^2) d\tau \end{cases} \quad (14)$$

式中, (x_0, y_0) 为起点, θ_0 为切线角初始值, k 为曲率, 下标 0 代表初始值, c 为曲率锐度参数, s 为弧长。 其中曲率 k 和切线角 θ 的计算如式 (15) 所示:

$$\begin{cases} k = k_0 + cs \\ \theta = \theta_0 + k_0 s + \frac{1}{2} cs^2 \end{cases} \quad (15)$$

假设算法中关键点坐标为 $P(x_i, y_i) (i = 1, 2, \dots)$ 上的离散点, 那么对各点曲线拟合的过程, 即保证曲率连续, 求解 k 个离散点的各曲线段。 图 3 为一组离散点经拟合后的曲线段。

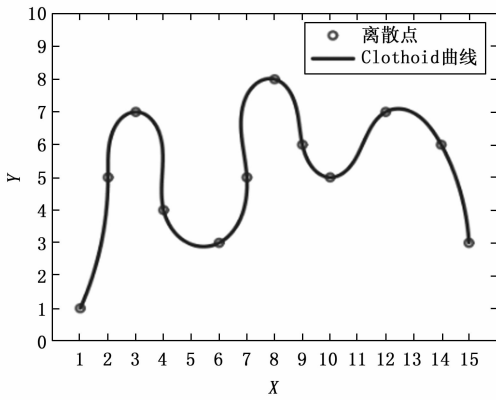


图 3 若干离散点拟合 clothoid 曲线段

对于任意的第 l 段曲线，其端点坐标满足如式 (16) 条件：

$$\begin{cases} x_{l+1} = x_l + \int_0^{s_l} \cos(\theta_{0l} + k_{0l}\tau + \frac{1}{2}c_l\tau^2) d\tau \\ y_{l+1} = y_l + \int_0^{s_l} \sin(\theta_{0l} + k_{0l}\tau + \frac{1}{2}c_l\tau^2) d\tau \end{cases} \quad (16)$$

式中，第 l 段弧长为 s_l ， (x_l, y_l) 点的切线角为 θ_{0l} ，曲率为 k_{0l} 。为保证各点曲率连续且平滑，各参数满足如式 (17) 的要求：

$$\begin{cases} k_{0l+1} = k_l = k_{0l} + c_l s_l \\ \theta_{0l+1} = \theta_l = \theta_{0l} + k_{0l} s_l + \frac{1}{2} c_l s_l^2 \end{cases} \quad (17)$$

式中，前后两段曲线交点处切线角、曲率均前后相等。即第 $l+1$ 段曲线起点切线角等于第 l 段曲线的终点切线角，第 $l+1$ 段曲线起点曲率等于第 l 段曲线的终点曲率。

2.5 改进算法描述

改进后的算法流程图大致可分为初始化阶段、网络建模阶段、判断阶段和位置更新阶段，具体如图 4 所示。

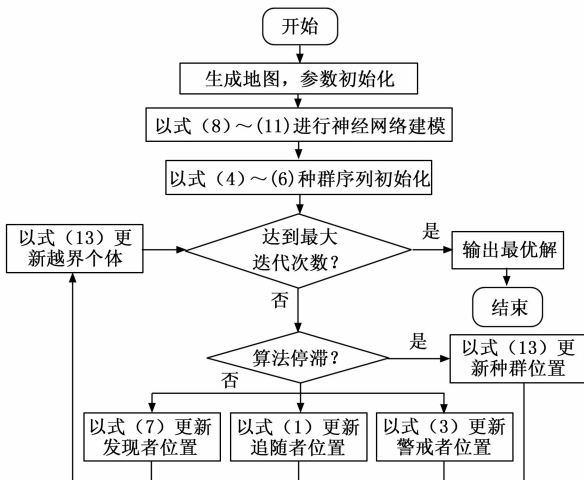


图 4 改进算法流程图

首先通过式 (8) ~ (11) 以三层神经网络对规划环境进行栅格化建模，生成膨胀后的栅格图，再对各参数初始

化；用公式 (4) ~ (6) 对种群进行低差异序列初始化，形成分布均匀的种群；分别以式 (1)、(3)、(7) 更新尾随麻雀、警戒麻雀和发现者的位置，若果算法陷入局部最优则以布朗运动步长通过式 (13) 跳出局部最优，若存在个体位置超过解空间上下界，也通过式 (13) 进行位置初始化；最后当满足设定迭代次数时，结束算法，再对路径进行平滑，同时输出平滑前后的路径，得到对应收敛曲线。

2.6 算法复杂度分析

时间复杂度反映了算法面对复杂问题的求解能力。若在 D 维解空间存在规模为 N 的初代种群，求解该问题需要消耗 $f(D)$ 的时间。则时间复杂度可表示为：

$$T = O(D + f(D)) \quad (18)$$

在一系列的改进后，初始化参数用时记为 η_0 ，各随机因子生成时间记为 η_1 ，任意一维种群产生耗时为 η_2 ，越界重置位置时间记为 η_3 ，此时的时间复杂度可表示为如下公式：

$$T_1 = O(\eta_0 + Nf(D) + D(\eta_1 + \eta_2 + \eta_3)) = O(D + f(D)) \quad (19)$$

种群存在不同类型个体。发现者有 $r_1 N$ 个， r_1 即发现者在种群所有个体中的占比，任意一维位置更新耗时 η_4 ，随机因子生成需消耗时长 η_5 ，则发现者执行阶段时间复杂度可表示为：

$$T_2 = O(r_1 N(\eta_4 + \eta_5)D) = O(D) \quad (20)$$

追随者有 $r_2 N$ ，同样的 r_2 为其在个体总量的占比，任意一维位置更新耗时 η_6 ，随机因子消耗时长 η_7 ，则追随者执行阶段时间复杂度可表示为：

$$T_3 = O(r_2 N(\eta_6 + \eta_7)D) = O(D) \quad (21)$$

除去发现者与追随者，剩余的均为警戒者，数量则为 $(1 - r_1 - r_2) N$ ，任一维位置更新耗时 η_8 ，随机因子消耗时长为 η_9 ，则警戒者执行阶段时间复杂度可表示为：

$$T_4 = O((1 - r_1 - r_2)N(\eta_8 + \eta_9)D) = O(D) \quad (22)$$

在布朗运动阶段，按式 (13) 更新麻雀最优个体所需的时间为 η_{10} ，生成随机步长 R 所需的时间分别为 η_{11} ，则此阶段的时间复杂度为：

$$T_5 = O(N(\eta_{10} + \eta_{11} + \eta_{12})D) = O(D) \quad (23)$$

最大迭代次数为 $iter_{max}$ ，改进算法的时间复杂度为：

$$T' = T_1 + iter_{max}(T_2 + T_3 + T_4 + T_5) = O(D + f(D)) \quad (24)$$

通过对比可知，算法改进前后的时间复杂度在同一水平尺度，优化改进得到的性能优化没有导致时间复杂度的额外增加。

3 仿真及分析

3.1 算法性能测试

选取 2 个单峰，3 个复杂多峰，1 个定维函数来测试算法寻优性能，具体函数如表 1 所示。同时将标准 SSA，GWO 和粒子群算法也加入一起作为横向对比，将文献 [20] 中的 ISSA 与文献 [21] 中的 CSSA 进行纵向对比，以

表 1 测试函数表

函数	维度(d)	区间	最小值
$F_1(x) = \sum_{i=1}^n (\sum_{j=1}^i X_j)^2$	30	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
$F_3(x) = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	30	$[-500, 500]$	-418.982 9 n
$F_4(x) = \sin^2(\pi\omega_1) + \sum_{i=1}^{d-1} (\omega_i - 1)^2 [1 + 10\sin^2(\pi\omega_1 + 1) + (\omega_d - 1)^2] [1 + \sin^2(2\pi\omega_d)]$	30	$[-10, 10]^d$	0
$F_5(x) = \sum_{i=1}^{d-1} (100x_i^2 - x_{i+1})^2 + (x_i - 1)^2$	30	$[-10, 10]^d$	0
$F_6(x) = (0.002 + \sum_{i=1}^{25} (\frac{1}{i + (x_1 - a_{1i})^6 + (x_2 - a_{2i})^6})^{-1})^{-1}$ where $a = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \dots & 32 & 32 & 32 \end{pmatrix}$	2	$[-65.536, 65.536]$	0.998

此突出本文改进算法的新颖性与优越性。对比算法种群数量统一为 100，迭代次数为 500，粒子群算法中的独有参数 $c1=c2=2$ ， $\omega=0.728$ 。统计各算法的平均值指标，并将各算法最终指标进行排序，根据排序先后衡量算法的寻优能力的强弱。表 2 中可以发现，改进算法对各函数均表现出较强的寻优能力，特别是在 $F_3 \sim F_6$ 中，改进算法的寻优效果显著。为避免均值指标的片面性，引入 Wilcoxon 秩检验，在 5% 的显著水平下进行试验，当 $P \leq 0.05$ 时，则表示存在显著性差异。反之当 $P > 0.05$ 时，则表示差异不显著，用 N/A 表示，具体结果如表 1 所示。由表 1 中看出，改进算法与其他各算法存在显著差异，改进算法更具有快速处理复杂问题的优势。

表 2 算法寻优结果排序

函数	PSO	GWO	SSA	ISSA	CSSA	本文改进
$F_1(x)$	6	5	1	4	1	1
$F_2(x)$	5	6	3	4	2	1
$F_3(x)$	6	5	2	3	4	1
$F_4(x)$	6	5	2	4	3	1
$F_5(x)$	6	5	2	4	3	1
$F_6(x)$	3	4	6	5	2	1

表 3 Wilcoxon 秩和检验 P 值

CSSA	F	PSO	GWO	SSA	ISSA
F_1	3.01×10^{-11}	3.01×10^{-11}	0.001 5	1.44×10^{-4}	2.82×10^{-4}
F_2	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F_3	3.01×10^{-11}	3.01×10^{-11}	0.001 3	0.003 0	N/A
F_4	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F_5	1.21×10^{-12}	1.21×10^{-12}	1.65×10^{-11}	1.21×10^{-12}	4.57×10^{-12}
F_6	N/A	2.70×10^{-11}	0.018 7	N/A	2.09×10^{-4}

3.2 地图环境生成

用本文改进算法与传统麻雀算法、文献 [19] 中的 CS-SA 算法作仿真结果对比。仿真计算机配置采用 intel i7-11800H 型号的 CPU，GeForce RTX 3070 显卡，32 GB 运行内存。

本文以膨胀法将环境进行栅格化建立环境模型。每个栅格代表一个节点，每次可移动一个单元格，因此有 8 个移动方向，如图 5 所示。

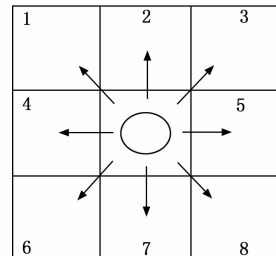


图 5 运动方向

通过建立两种图 5 所示的不同栅格环境，验证算法在不同尺度环境下的移动机器人的实际规划效果。栅格地图中，黑色部分即障碍，白色部分为可规划区域。坐标 (0.5, 0.5) 为规划起点，坐标 (19.5, 19.5) 和 (39.5, 39.5) 为规划终点，在图 6 以灰色栅格表示，在仿真过程中黑色用 1 表示，白色和灰色用 0 表示。

栅格图中原点位于左下角，终点在右上角第 j 行 k 列，代号为 G 的栅格中心坐标为：

$$\begin{cases} x = \text{mod}(G, j) - 1/2 \\ y = \text{int}(G, k) + 1/2 \end{cases} \quad (25)$$

式中，横、纵坐标即 x 和 y ， mod 为取余运算， int 为求整公式。

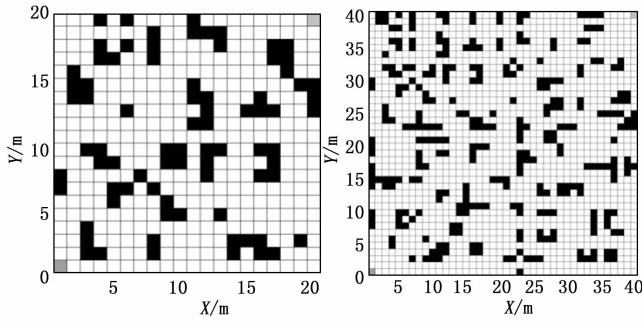


图 6 环境栅格图

3.3 算法参数

通过 Matlab 软件得到 20×20 与 40×40 栅格图，通过比较测试 SSA 算法、CSSA 算法、本文改进算法在栅格地图上规划出的最短路径来对比改进算法的优越性。其主要参数设置如表 4 所示。

表 4 基本参数设置

参数	含义	参数值
PD	发现者比例	$0.2 N$
SD	侦察者比例	$0.8 N$
R_s	警戒阈值	0.8
ST	安全阈值	0.6
N	种群规模	100
$iter_{max}$	最大迭代次数	200

3.4 仿真分析

在环境大小为 20×20 的栅格地图中对 3 种算法进行仿真，得到的最优路径对比和收敛迭代曲线对比如图 7 所示。

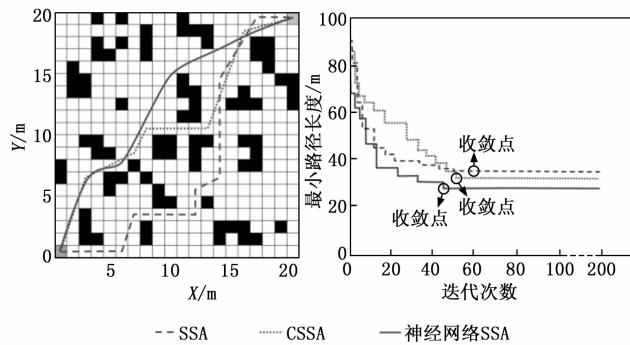


图 7 20×20 地图仿真路径与收敛曲线对比图

图 7 中，左图为改进算法、原始算法同对比算法规划出的路径，改进算法因为引入了平滑改进，所以相对于其他两种算法没有尖锐转弯点，而原始算法则有 7 个尖锐转弯点，对比算法也有相应优化，只有 5 个尖锐转弯点。右图为 3 种算法的收敛曲线，原始算法迭代 60 次后趋于收敛，对比算法迭代 52 次后收敛稳定，改进算法最优，46 代即收敛于最优路径。

在环境大小为 40×40 的栅格地图中对 3 种算法进行仿真，得到的最优路径对比和收敛迭代曲线对比如图 8 所示。

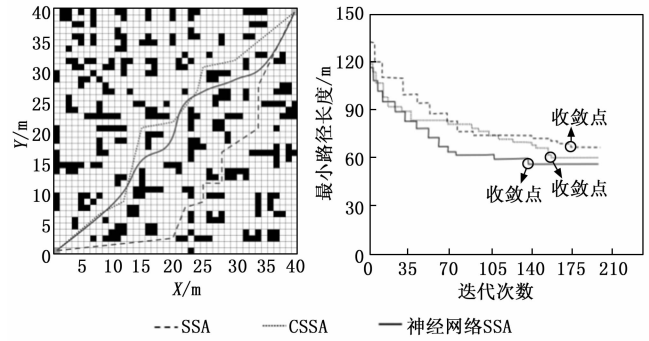


图 8 40×40 地图仿真路径与收敛曲线对比图

图 8 中，左图为改进算法、原始算法同对比算法规划出的路径，改进算法因为引入了平滑改进，所以相对于其他两种算法没有尖锐转弯点，由于地图尺度增加，环境障碍更加复杂，原始算法有 8 个尖锐转弯点，对比算法只有 7 个尖锐转弯点。右图为 3 种算法的收敛曲线，原始算法迭代 175 次后趋于收敛，对比算法迭代 158 次后收敛稳定，改进算法最优，139 代即收敛于最优路径。

在 20×20 地图和 40×40 地图中，对比传统 SSA、改进 SSA 和 CSSA 三种算法规划出的路径，传统 SSA 算法和 CSSA 算法由于没有路径平滑过程，均有较多的尖锐拐点，且收敛速度相对改进 SSA 算法更慢，其中传统 SSA 算法路径相对较长，加大了机器人不必要的能量消耗。改进 SSA 算法在路径长度上最短，收敛速度最快，且无尖锐拐点。具体指标对比如表 5 所示。

表 5 指标对比

环境大小	算法	路径长度/m	收敛迭代次数/次	尖锐拐点个数/个
20×20	SSA	34.38	60	7
	改进 SSA	28.31	46	0
	CSSA	30.67	52	5
40×40	SSA	65.98	175	8
	改进 SSA	55.80	139	0
	CSSA	59.83	158	7

结果显示，在 20×20 地图环境下，改进 SSA 相对传统 SSA 算法最短路径缩短了 6.07 m，降低了 17.66%，收敛迭代次数减少 14 次约 23.33%；改进 SSA 相对 CSSA 算法最短路径缩短了 2.36 m，降低了 7.69%，收敛迭代次数减少 6 次约 11.54%。在 40×40 地图环境下，改进 SSA 相对传统 SSA 算法最短路径缩短了 10.18 m，降低了 15.43%，收敛迭代次数减少 36 次约 20.57%；改进 SSA 相对 CSSA 算法最短路径缩短了 4.03 m，降低了 6.74%，收敛迭代次数减少 19 次约 12.03%。

综合来看，改进 SSA 算法在不同尺度环境解决机器人避障问题表现均很突出，其各项性能指标均优于 CSSA 算法和传统 SSA 算法，可以以最快的速度收敛于最短平滑路径。

(下转第 288 页)