文章编号:1008-1542(2023)02-0165-12

基于改进的 DDPG 算法的蛇形机器人路径规划方法

郝崇清[1],任博恒[1],赵庆鹏[2],侯宝帅[1],白　彤[1],武晓晶[1],樊劲辉[1]

(1.河北科技大学电气工程学院,河北石家庄　050018;2.南京邮电大学通信与信息工程学院,江苏南京　210023)

摘　要:针对蛇形机器人执行路径规划任务时,面对复杂环境传统强化学习算法出现的训练速度慢、容易陷入死区导致收敛速度慢等问题,提出了一种改进的深度确定性策略梯度(deep deterministic policy gradient,DDPG)算法。首先,在策略-价值(actor-critic)网络中引入多层长短期记忆(long short-term memory,LSTM)神经网络模型,使其控制经验池中信息的记忆和遗忘程度;其次,通过最优化特征参数将 CPG(central pattern generators)网络融入强化学习模型,并设计新型网络状态空间和奖励函数;最后,将改进算法与传统算法分别部署在 Webots 环境中进行仿真实验。结果表明,相比于传统算法,改进算法整体训练时间平均降低了 15%,到达目标点迭代次数平均降低了 22%,减少了行驶过程中陷入死区的次数,收敛速度也有明显的提升。因此所提算法可以有效地引导蛇形机器人躲避障碍物,为其在复杂环境下执行路径规划任务提供了新的思路。

关键词:机器人控制;蛇形机器人;改进的 DDPG 算法;强化学习;CPG 网络;Webots 三维仿真

中图分类号:TP242.6　　**文献标识码**:A　　**DOI**:10.7535/hbkd.2023yx02007

Path planning method of snake-like robot based on improved DDPG algorithm

HAO Chongqing[1], REN Boheng[1], ZHAO Qingpeng[2], HOU Baoshuai[1], BAI Tong[1], WU Xiaojing[1], FAN Jinhui[1]

(1. School of Electrical Engineering, Hebei University of Science and Technology, Shijiazhuang, Hebei 050018, China; 2. School of Communication and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210023, China)

收稿日期:2022-12-21;修回日期:2023-02-25;责任编辑:冯　民

基金项目:国家自然科学基金(62003129);河北省重点研发计划项目(20326628D)

第一作者简介:郝崇清(1981—),男,山东枣庄人,副教授,博士,主要从事机器视觉、智能仿生机器人方面的研究。

通信作者:樊劲辉副教授。E-mail: fanhebust@163.com

郝崇清,任博恒,赵庆鹏,等.基于改进的 DDPG 算法的蛇形机器人路径规划方法[J].河北科技大学学报,2023,44(2):165-176.

HAO Chongqing, REN Boheng, ZHAO Qingpeng, et al. Path planning method of snake-like robot based on improved DDPG algorithm[J]. Journal of Hebei University of Science and Technology, 2023, 44(2): 165-176.

Abstract: Aiming at the problems of low training speed and convergence speed caused by falling into a dead zone of traditional reinforcement learning algorithm of the snake-like robot when performing path planning task in multi-obstacle environment, an improved deep deterministic policy gradient (DDPG) algorithm was proposed. Firstly, a multi-layer long short-term memory (LSTM) neural network model was introduced into the actor-critic network to control the memory and forgetting degree of information in the experience pool; secondly, the CPG (central pattern generators) network was integrated into a reinforcement learning model by optimizing feature parameters, designing new network state space and reward function, finally, The improved algorithm and the traditional algorithm were deployed in Webots environment for simulation experiments. The results show that compared with the traditional algorithm, the overall training time of the improved algorithm is reduced by 15% on average, and the number of iterations to reach the target point is reduced by 22% on average, which reduces the times of falling into the dead zone during driving and obviously improves the convergence speed. The algorithm can effectively guide the snake-like robot to avoid obstacles, thus providing a new idea for its performing path planning task in multi-obstacle environment.

Keywords: robot control; snake-like robot; improved DDPG algorithm; intensive learning; CPG network; Webots 3D simulation

蛇形机器人因其强大的环境适应能力,被广泛应用于地质勘探、灾后救援和医疗等领域[1]。在执行任务时,蛇形机器人需要结合多种环境信息进行路径规划和导航,其中路径规划作为导航的主要部分,其结果的优劣程度直接影响了蛇形机器人完成任务的质量[2]。然而,传统的路径规划方法都需要部署在已知环境下,例如A*算法[3]、卡尔曼滤波算法[4]、LOS(line of sight)算法[5]等,面对未知障碍物场景,怎样引导其自主完成路径规划任务成为蛇形机器人研究的热点话题。

近年来,随着人工智能算法的发展,强化学习(reinforcement learning,RL)算法[6]在蛇形机器人路径规划中得到了广泛的应用。该算法通过与环境进行交互从而优化蛇形机器人的动作,使其自主完成路径规划任务,但是随着环境复杂度的增加,导致RL算法收敛速度慢,难以处理高维连续状态和动作信息。为了解决该类问题,JIA等[7]提出了一种基于改进RL算法的蛇形机器人路径规划方法,该方法将蛇形机器人运动学模型融入RL算法中,实现路径规划和避障的同时,有效节省了算法收敛时间,减少了发散次数。BING等[8]将RL算法与逆强化学习(inverse reinforcement learning,IRL)算法相结合提出了一种节能和损伤恢复的滑动步态方法,通过设计相关控制器,使其生成自适应运动步态的同时拥有损伤恢复能力。

利用RL算法进行路径规划相关任务时,为了降低蛇形机器人连续状态空间的信息维度,提高算法的训练速度,LIU等[9]提出了一种基于RL算法的软体机器蛇目标跟踪控制方法,将CPG(central pattern generators,CPG)网络与强化学习模块相结合,通过对松冈CPG系统振荡特性的理论分析,利用强化学习算法在模拟环境中学习控制策略,以用于蛇机器人执行目标跟踪任务,该方法不仅使蛇形机器人具备了一定的环境自适应能力,而且还降低了RL算法的信息传输维度。严浙平等[10]提出了一种基于模型预测和中枢模式发生器的轨迹跟踪控制方法,用于控制六足仿生机器人,该方法在实验中表现出良好的运动性能和稳定性。

针对多种障碍物场景,蛇形机器人需要更强的自适应路径规划能力,从而避免陷入死区导致目标不可达的问题,QIN等[11]提出了一种欠驱动的自适应轨迹控制方法,使蛇形机器人能够跟踪到较小的工作空间。JIANG等[12]结合RL算法提出了一种具有不同拓扑结构的多层脉冲神经网络(SNN)模型,从蛇形机器人动态视觉传感器(DVS)获得视觉信号,驱动其运动控制器跟踪特定的运动对象,以便快速地走出死区区域。为了提高RL算法的收敛速度、降低网络训练时间,张瀚等[13]将深度确定性策略梯度算法(DDPG)与人工势场法相融合,有效提升了算法的性能。CHO等[14]利用DDPG算法进行蠕蟒机器人的路径规划,使用Gazebo动态模拟器设置可移动障碍物,并通过算法训练模型使蠕蟒机器人能顺利完成自适应路径规划任务,验证了DDPG算法的稳定性。国内外学者虽然提出了功能更强大的RL算法,但是面对多障碍环境仍然存在算法随机性强、容易陷入死区状态,造成算法训练时间长、收敛速度慢等问题。

针对上述问题,通过在策略-价值网络中引入多层LSTM神经网络模型,使DDPG算法拥有选择性记忆功能,并根据蛇形机器人复杂的运动环境,设计全新的状态空间和相关奖励函数,从而引导DDPG算法更好地完成算法训练和路径规划任务。为了减少DDPG算法中数据的传输维度,利用Matsuoka振荡器搭建CPG网络并且最优化超参数,将CPG网络融入DDPG算法中用于实现蛇形机器人多模式运动。为了验证

算法的有效性，在 Webots 环境中搭建了蛇形机器人模型，将所提算法部署在该模型上进行路径规划实验研究，并对该算法在不同环境复杂度下的性能进行分析。

1　强化学习算法

1.1　深度确定性策略梯度算法[15]

确定性策略梯度算法(deterministic policy gradient，DPG)[16]作为强化学习的经典算法，可以完成高维且连续的运动空间任务，但是因其动作的不确定性导致网络输出具有很强的随机性。该算法通过引入价值网络进行判断，遍历蛇形机器人运动中每个状态 S_t 下的所有动作值，使其在连续运动中根据概率分布函数 π 在每次迭代次数下进行采样，从而获得当前迭代次数下的最优动作 a_t^*，为了使蛇形机器人得到价值更高的动作作为网络输出，通过函数 μ 在最优动作 a_t^* 中选择最大概率动作 $\bar{a}_t$。

$$a_t^* \sim \pi_\theta(s_t \mid \theta^\pi)\ , \tag{1}$$

$$\bar{a}_t \sim \mu_\theta(s_t \mid \theta^\mu)\ 。 \tag{2}$$

为了降低 DPG 算法网络的随机性，DDPG 算法将 DPG 算法与深度神经网络(deep neural networks，DNN)[17]相结合，利用 Actor-Critic 网络作为核心架构并根据蛇形机器人获取的当前环境信息，从而将其路径规划问题转化成马尔可夫决策过程，随着迭代次数的增加，根据马尔可夫决策序列 $\{S_0, A_0, R_1, S_1, A_1, R_2, \cdots, S_{t+1}, A_{t+1}, R_t, S_t, A_t\}$ 使蛇形机器人得到随机性策略和每次迭代的回报值。

DDPG 算法框架如图 1 所示。S_t, A_t, R_t 分别表示当前网络迭代下机器人的状态 s、动作 a 以及上轮网络迭代得到的奖励值。蛇形机器人通过价值网络评估从而选择最优动作策略，使网络获得最大回报值，其定义如式(3)所示。

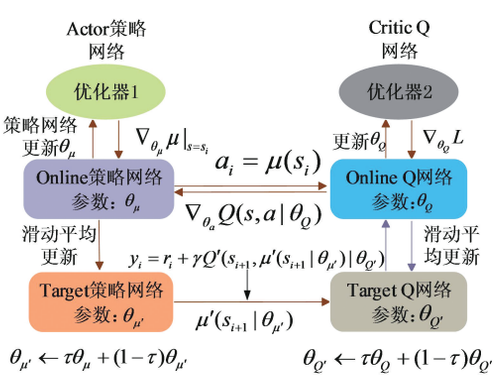


图 1　DDPG 算法框架示意图

Fig. 1　Schematic diagram of DDPG algorithm framework

$$Q_t(\mu)=R_{t+1}+\gamma R_{t+2}+\cdots=\sum_{\tau=0}^{+\infty}\gamma^\tau R_{t+\tau+1}\ , \tag{3}$$

式中：γ 表示价值折扣因子；Q_t 表示累计价值回报值。DDPG 算法为了搜索一个合适的参数 θ 使策略函数 π 最优，利用卷积神经网络搭建 Actor 网络 $\mu(s)$ 和 Critic 网络 $Q(s,a)$，通过蛇形机器人与环境进行交互，从而直接进行端对端的学习。为了提高 DDPG 算法的学习效率和稳定性，使用经验回放机制打破训练数据的相关性，在算法生成样本训练数据时，将马尔可夫决策序列存放在记忆池中，使其每次更新 Actor-Critic 网络时都会进行样本优化，其中 Actor-Critic 网络包括当前更新网络 $Q(s,a\mid\theta_Q)$ 和 $\mu(s\mid\theta_u)$，以及目标网络 $Q'(s,a\mid\theta_{Q'})$ 和 $\mu'(s\mid\theta_{u'})$。采用 TDerror 的方式对 Critic 网络进行更新[15]：

$$\begin{cases} L=\dfrac{1}{N}\sum_i(y_i-Q(s_i,a_i\mid\theta_Q))^2\ , \\ y_i=r_i+\gamma Q'(s_{i+1},\mu'(s_{i+1}\mid\theta_{\mu'})\mid\theta_{Q'})\ , \end{cases} \tag{4}$$

式中：L 表示损失函数的最小方差值；N 表示选取的样本数量；θ 表示网络参数。蛇形机器人通过获得上轮网络迭代的状态 S_{t+1} 和 Actor 网络的输出 $\mu'(s|\theta_{\mu'})$ 使其得到目标网络的输出 $Q'(s,a|\theta_{Q'})$，随着迭代次数的增加，利用 $\mu'(s_{i+1}|\theta_{\mu'})$ 计算 $Q'(s_{i+1},\mu'(s_{i+1}|\theta_{\mu'})|\theta_{Q'})$ 从而预测下轮迭代的网络输出，其中 Critic 网络 Q 的输出用于计算网络损失函数 L，y_i 用于计算 TDerror 目标。为了获得更好的网络参数 θ_Q 更新 Critic 网络，利用反向传播方法针对 θ_Q 求梯度 $\nabla_\theta L$，根据蛇形机器人当前状态 S_t 和 Critic 网络输出 $Q(s,a|\theta_Q)$ 进行 Actor 网络的更新[15]，即：

$$\nabla_{\theta_\mu}\mu\mid_{s=s_i}=\frac{1}{N}\sum_i\nabla_{\theta_a}Q(s,a\mid\theta_Q)\mid_{s=s_i,a=\mu(s_i)}\nabla_{\theta_\mu}\mu(s\mid\theta_\mu)\mid_{s=s_i}\ 。 \tag{5}$$

目标网络的参数更新采用滑动平均的方法：

$$\begin{cases} \theta_{Q'}\leftarrow\tau\theta_Q+(1-\tau)\theta_{Q'}\ , \\ \theta_{\mu'}\leftarrow\tau\theta_\mu+(1-\tau)\theta_{\mu'}\ 。 \end{cases} \tag{6}$$

1.2 LSTM 神经网络模型

随着迭代次数的不断增加，DDPG 算法总会削弱之前得到的最优经验影响。在 DNN 算法中 LSTM 神经网络模型[18]可以控制网络对于经验的记忆和遗忘程度，其网络模型是基于循环神经网络模型(recurrent neural network，RNN)的一种改进。与传统 RNN 算法相比，LSTM 算法在其隐藏层中增加了一个细胞状态(cell state)[19]，使网络自循环的权重产生了变化，在模型参数固定的情况下，根据网络权重的动态改变可以调节不同时刻网络更新状态，从而避免梯度消失或梯度爆炸的问题。LSTM 算法包含的 3 种门结构分别定义为记忆门、遗忘门和输出门，其网络结构如图 2 所示。

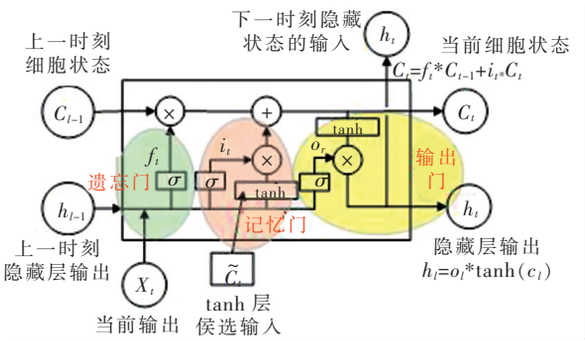


图 2 LSTM 神经网络算法结构图

Fig. 2 Structure of LSTM neural network algorithm

单个 LSTM 神经网络模型转化为细胞状态公式[18]：

$$\begin{cases} f_t=\sigma(\boldsymbol{W}_f\cdot[h_{t-1},x_t]+b_f), \\ i_t=\sigma(\boldsymbol{W}_i\cdot[h_{t-1},x_t]+b_i), \\ \tilde{\boldsymbol{C}}_t=\tanh(\boldsymbol{W}_c\cdot[h_{t-1},x_t]+b_c), \\ \boldsymbol{C}_t=f_t*\boldsymbol{C}_{t-1}+i_t*\tilde{\boldsymbol{C}}_t, \\ o_t=\sigma(\boldsymbol{W}_o\cdot[h_{t-1},x_t]+b_o), \\ h_t=o_t*\tanh(\boldsymbol{C}_t), \end{cases} \tag{7}$$

式中：x_t 表示网络当前输入；f_t 表示遗忘门限；i_t 表示输入门限；o_t 表示输出门限；σ 表示激活函数；$\boldsymbol{W}$ 表示当前门的网络权重；b 表示当前门的偏置参数；$\tilde{\boldsymbol{C}}_t$ 表示候选输入向量；$\boldsymbol{C}_t$ 表示当前时刻网络状态；$\boldsymbol{C}_{t-1}$ 表示前一时刻网络状态；h_{t-1} 表示上一时刻隐藏层输出。

2 蛇形机器人路径规划方法

2.1 改进的 DDPG 算法

在进行蛇形机器人路径规划任务时，DDPG 算法容易出现陷入死区问题，不仅降低算法训练速度和收敛速度，而且会导致任务无法完成。为了提高算法的稳定性，在 DDPG 算法的 Actor-Critic 网络中引入多层 LSTM 神经网络模型，将 Actor-Critic 网络全连接层替换成 LSTM 神经元，通过滑动窗口控制样本信息的记忆和遗忘程度，使其优先学习高奖励值的动作。面对复杂环境，为提高蛇形机器人的适应能力，根据环境设计全新的状态空间和奖励函数，从而更快地进行奖励累计并训练更好的模型。

改进的 DDPG 算法将 Actor-Critic 网络的前 3 个全连接层替换成多层 LSTM 神经网络模型，为了防止网络反向传播过程中出现梯度消失问题，利用全连接层在整个神经网络模型中将样本特征映射到标记空间中，并与网络上轮迭代获得的隐藏状态一同送入 LSTM 单元中。为了获得所需要的期望运动空间，将 LSTM 单元提取到的高维特征映射至一维向量空间内，通过之前提取的特征语义信息，进行输出降维并送入连接层中。

根据蛇形机器人的路径规划任务以及雷达、视觉传感器的数据信息，改进的 DDPG 算法将蛇形机器人当前网络迭代下所获得的状态空间信息和采取的动作分别作为输入和输出，其中每轮迭代获得的动作根据 Critic 网络评判都会获得相应的奖励值，通过 Actor 网络和 Critic 网络进行梯度更新，从而获得更新后的网络权重以及训练模型。改进的 DDPG 算法中 Actor 网络和 Critic 网络分别由 4 层神经网络构成，其网络结构如图 3 所示，Actor-Critic 网络前 3 层为 LSTM 记忆单元，第 4 层为全连接层，其每个记忆单元包括了 256 个隐藏神经元节点，分别利用 sigmoid 函数和 tanh 函数作为激活函数，通过获取蛇形机器人状态空间的时序信息，获得其转弯角度和运动方向。

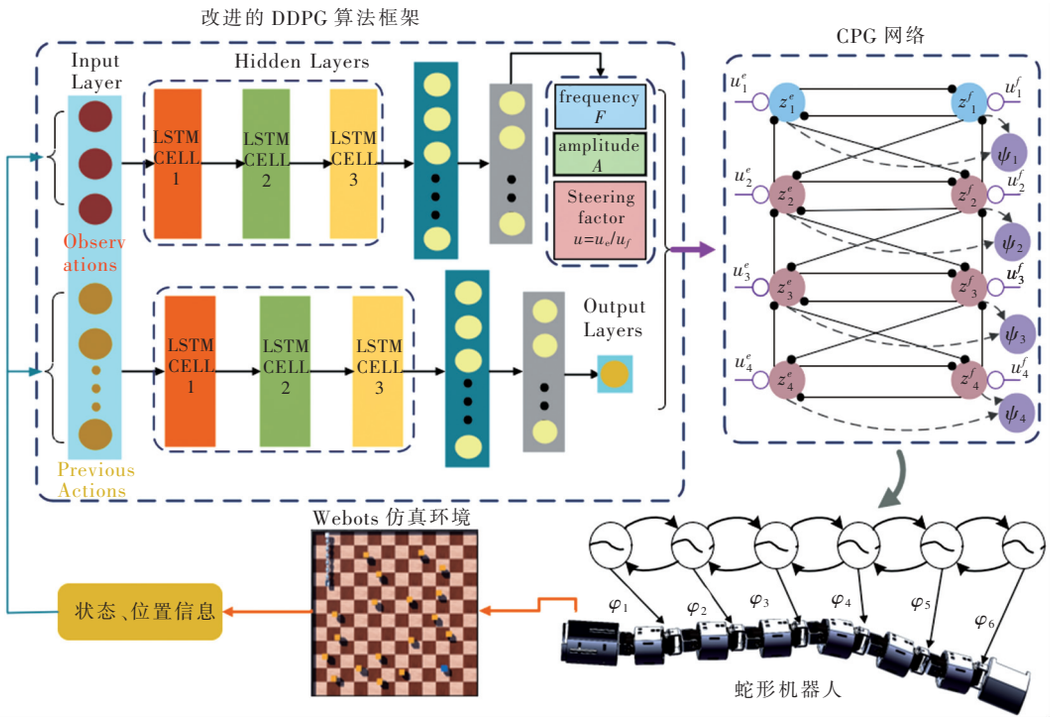


图 3　蛇形机器人路径规划任务模型训练架构示意图

Fig. 3　Schematic diagram of path planning task model training for a snake robot

2.2　蛇形机器人 CPG 控制网络

为了使蛇形机器人转弯和运动方式更贴合生物蛇，本文对蛇形机器人关节进行了运动学建模，蛇形机器人样机如图 4 a)所示，关节结构如图 4 b)所示。其蜿蜒运动中，根据蛇形机器人运动方式构建其全局坐标系 $W_w=\{O_w\text{-}A_w,\ B_w,C_w\}$ 并设置纵向关节为静关节，使 CPG 网络的输出作为横向关节运动信号。根据蛇形机器人关节运动模型，构建相邻两点的旋转变化矩阵 ${}^{w}\boldsymbol{D}_i$ 和关节坐标系 $C_i=\{O_i\text{-}X_i,Y_i,Z_i\}$，并对第 $(i-1)$ 关节位置向量 ${}^{i-1}\boldsymbol{p}_i$ 进行表示，如式(8)所示[11,20]：

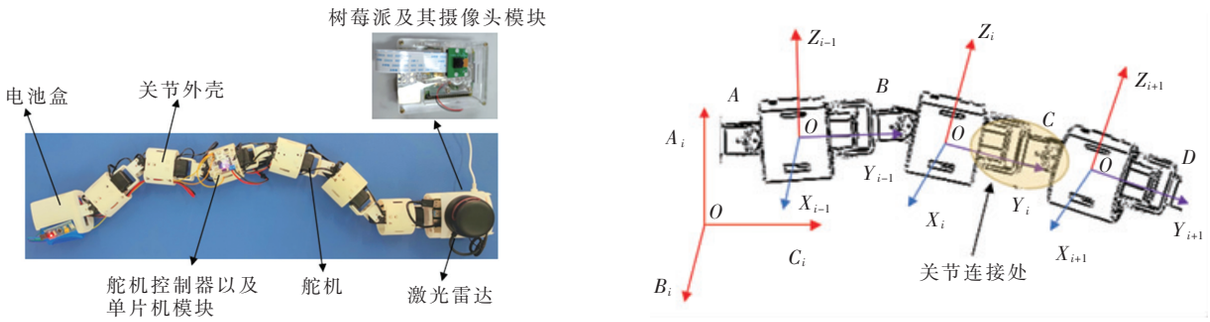
$$\begin{cases} {}^{w}\boldsymbol{D}_i={}^{A}_{B}\boldsymbol{D}={}^{B}_{C}\boldsymbol{D}={}^{C}_{D}\boldsymbol{D}= \\ \qquad \boldsymbol{R}(x,\theta_i)\mathrm{Trans}(0,2h+l,0)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_i) & -\sin(\theta_i) & (2h+l)\cos(\theta_i) \\ 0 & \sin(\theta_i) & \cos(\theta_i) & (2h+l)\sin(\theta_i) \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\ {}^{i-1}\boldsymbol{p}_i=\begin{bmatrix} l_{i-1} \\ 0 \\ 0 \end{bmatrix}, \end{cases} \tag{8}$$

式中：l_{i-1} 表示蛇形机器人关节质心长度；h 表示关节连接处长度；θ 表示关节绕全局坐标系的旋转角度。根据蛇形机器人关节变化矩阵 ${}^{w}\boldsymbol{D}_i$，通过将原点 O_w 设为蛇形机器人初始运动点，从而得到其平均速度矩阵[18] ${}^{w}\boldsymbol{V}_i=[\boldsymbol{V}_{x,i}\boldsymbol{V}_{y,i}\boldsymbol{V}_{z,i}]^{\mathrm{T}}$，如式(9)所示[20]：

$$\begin{cases} {}^{w}\boldsymbol{r}_i={}^{w}\boldsymbol{D}_i\cdot{}^{j}\boldsymbol{r}_i+{}^{w}\boldsymbol{p}_i, \\ {}^{w}\boldsymbol{p}_i={}^{w}\boldsymbol{p}_0+\sum_{i=1}^{i}{}^{w}\boldsymbol{D}_{i-1}\cdot{}^{j-1}\boldsymbol{p}_i, \\ {}^{w}\boldsymbol{V}_i={}^{w}\dot{\boldsymbol{r}}_i={}^{w}\dot{\boldsymbol{D}}_i\cdot{}^{j}\dot{\boldsymbol{r}}_i+{}^{w}\dot{\boldsymbol{p}}_i, \end{cases} \tag{9}$$

式中：${}^{w}\boldsymbol{p}_0=[x_0\quad y_0\quad 0]^{\mathrm{T}}$；${}^{w}\boldsymbol{r}_i$ 为蛇形机器人头部起点与 W_w 的位置向量；${}^{j}\boldsymbol{r}_i$ 为蛇形机器人关节质心与 C_i 的位置向量；${}^{w}\boldsymbol{p}_i$ 为蛇形机器人关节坐标系 C_i 与 W_w 的位置向量；${}^{j-1}\boldsymbol{p}_i$ 为相邻两关节的位置向量矩阵。

根据蛇形机器人关节运动模型设计了 CPG 控制网络，该网络由互相连接的 Matsuoka 振荡器组成，其



a）蛇形机器人实物图　　b）蛇形机器人关节结构图

图 4　蛇形机器人实物及其关节示意图

Fig. 4　Snake robot prototype and joint diagrams

每个振荡器包含了一对互相抑制的神经元模型，其 CPG 网络数学模型如式(10)所示[9]：

$$\begin{cases} F\tau_r \dot{x}_i^e = -x_i^e - a y_i^f - b v_i^e - \sum_{j=1}^{N} w_{ji} y_j^e + u_i^e, \\ F\tau_a \dot{v}_i^e = y_i^e - v_i^e, \\ F\tau_r \dot{x}_i^f = -x_i^f - a y_i^e - b v_i^f - \sum_{j=1}^{N} w_{ji} y_j^f + u_i^f, \\ F\tau_a \dot{v}_i^f = y_i^f - v_i^f, \\ \varphi_i = A(y_i^e - y_i^f), \end{cases} \tag{10}$$

式中：e，f 分别表示伸肌和屈肌神经元的内部相关变量；$(x_i^{e,f}, v_i^{e,f})$表示第 i 个神经元的激活状态和内部抑制状态；$y_i^{e,f} = \max(0, x_i^{e,f})$表示单个屈伸肌神经元的输出；$a$ 表示屈肌和伸肌神经元连接的权重；b 表示单个神经元内激活和抑制状态的权重；$u_i^{e,f}$ 表示屈肌和伸肌神经恒定外部输入；w_{ji} 表示 2 个振荡器间的连接权重；$\tau_r, \tau_a \in R$ 表示神经元间的放电速率；A 表示屈伸肌之差的放大比；F 表示屈伸肌的频率比。

在神经元模型中，所有的负加权是抑制信号，正加权是激励信号。引入激活信号 u 作为 CPG 网络的输入，$u = [u_1^e, u_1^f, u_2^e, u_2^f, u_3^e, u_3^f, u_4^e, u_4^f, u_5^e, u_5^f, u_6^e, u_6^f]$。通过预先设置激活信号和超参数，从而使 CPG 网络持续输出理想的节律波。在蛇形机器人系统中，其蜿蜒运动需要 6 个关节协同运动，根据其刚体的实际要求，给 CPG 网络输出设定阈值 $\varphi_i \in [-1.75\ \text{rad}, 1.75\ \text{rad}]$，其中正范围表示振荡器伸肌神经被激活屈肌神经被抑制，负范围则与其相反。

2.3　路径规划模型训练框架

为了减少 DDPG 算法中输入、输出的数据维度并提高算法的训练效率，通过分析 CPG 网络模型参数并进行最优化处理，获取控制蛇形机器人运动的振幅 A、频率 F 和转向参数 u。如图 3 所示，将 CPG 网络模型与改进的 DDPG 算法框架和 Webots 仿真环境框架相结合，组成循环网络，即蛇形机器人自学习框架。在 Webots 仿真环境中根据视觉感知和全局定位坐标结合，获取障碍物的位置并作为 Actor 网络的输入，使其输出一个实时的确定性动作。为了获得一个最理想的动作作为输出，改进的 DDPG 算法利用 Critic 网络依据状态空间和设定的奖励去为当前动作打分。为了降低 Actor-Critic 网络的输入、输出维度，将 CPG 网络经过模块化处理，输入为最优特征参数，输出为速度、角速度、转向因子的大小，结合到 Webots 仿真的蛇形机器人模型中，从而实现在线实时控制。

2.4　状态、动作空间搭建与奖励函数设计

为了验证改进的 DDPG 算法在蛇形机器人路径规划任务中的有效性，利用蛇形机器人的 Solidworks 模型搭建了 Webots 仿真环境并将算法部署到仿真中进行训练。为了对比不同算法的实际效果，使用控制变量法，设置一致的网络参数用于更新训练模型，如表 1 所示。在 Webots 仿真中，黄色方块为障碍物模型，蓝色方块为蛇形机器人路径规划任务目标点，如图 5 所示。

表 1　模型训练参数设定

Tab. 1　Model training parameter setting

参　数	数值
衰减因子	0.95
稀疏环境训练次数	150
密集环境训练次数	400
测试次数	50
最大迭代次数	8 000
策略网络学习率	0.000 1
价值网络学习率	0.001
经验池最大容量	8 000

执行路径规划任务时，蛇形机器人每次网络迭代所获得的信息包括机器人的当前位置、关节角度变化、CPG 网络的最优特征参数、目标点距离和方位以及障碍物距离和方位。为了提高算法的收敛速度，减少训练时间，本文利用 CPG 网络的最优特征参数去控制机器蛇的运动并设计相应的状态空间 $S_t=(N,d_1\sim d_5,\theta_{d_n},\theta_{\text{end}},D_{\text{end}},P)$，如图 5 所示。$N$ 表示障碍物数量，θ_{d_n} 表示最近障碍物与蛇头的夹角，θ_{end} 表示目标点与蛇头的夹角，D_{end} 表示目标点与蛇头的距离，P 表示 CPG 网络最优特征参数。为了避免蛇形机器人碰撞障碍物，设置出现在摄像头中的障碍物数量 $N\leqslant 5$ 并根据距离设置阈值。为了评估运动是否接近目标，通过设置 $d_1\sim d_5$ 表示可视范围中最近 5 个障碍物与蛇头的相对距离。建立蛇形机器人运动学模型，如式(3)所示，定义动作空间为线速度 V 和偏移角度 φ，其运动参数包括线速度$[v_{\min},v_{\max}]$和蛇头偏移角度$[\varphi_{\min},\varphi_{\max}]$，并设置速度范围[0.0 m/s，0.2 m/s]和偏移角度范围[−1.5 rad，1.5rad]。

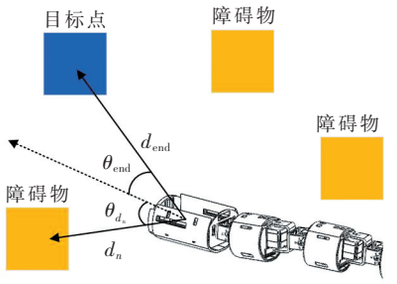


图 5　蛇形机器人动作空间示意图

Fig. 5　Action space diagram of a snake robot

奖励函数会影响强化学习的收敛性，根据奖励函数设置的优劣，从而使 Actor-Critic 网络得到更好的性能进行网络模型训练。本文设计的奖励函数由 3 方面组成，分别为蛇形机器人与环境障碍物距离信息、摄像头信息和关节角度，如式(11)所示：

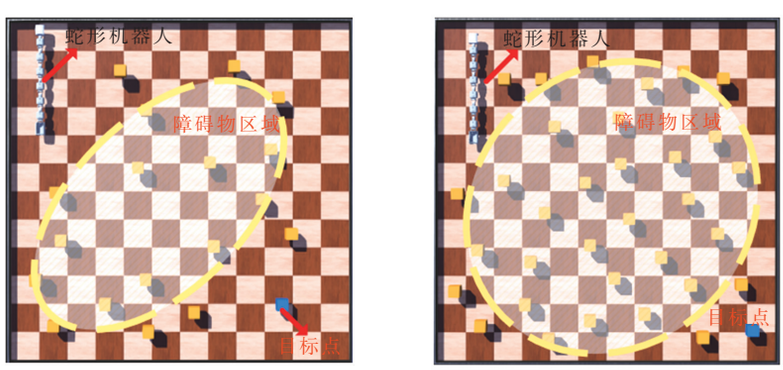
$$\begin{cases} R_1(s_t,a_t)=\begin{cases} r_1, & d_{l\sim 5}>D_{\text{m}},\\ r_2, & D_{\text{end}}<D_{\text{mh}},\\ r_3, & D_{\text{wall}}<0.2,\\ r_4, & \theta_{\text{end}},\theta_d\geqslant(n+1)\dfrac{\pi}{2}, \end{cases}\\ R_2(s_t,a_t)=\begin{cases} -100, & N\geqslant 5,\\ 100, & \text{others}, \end{cases}\\ R_3(s_t,a_t)=\begin{cases} -100, & \varphi>1.5\text{ rad or }\varphi<-1.5\text{ rad},\\ 100, & \text{others}, \end{cases} \end{cases} \tag{11}$$

式中：R_1 表示距离判断函数；R_2 表示摄像头信息判断函数，蛇形机器人的运动方向总是向障碍物少的一侧偏移；R_3 表示蛇头转动范围，其受到刚体结构限制；D_{m} 表示距离障碍物的最小距离；D_{wall} 表示与墙体的最小距离，避免蛇形机器人与障碍物和墙体产生碰撞，D_{mh} 表示与目标点的最小距离，用以指引蛇形机器人的运动方向。

3　仿真实验

3.1　仿真环境创建及参数设置

为了更加清晰地观察改进 DDPG 算法的收敛速度和训练效果，在 Webots 环境中，根据障碍物的密集程度搭建了 2 种仿真环境，分别为稀疏障碍物场景和密集障碍物场景。其中稀疏障碍物场景，地图大小为 3 m×3 m，障碍物大小为 25 cm×25 cm，数量为 20 个并随机分布，如图 6 a)所示。密集障碍物环境，地图大小为 3 m×3 m，障碍物大小为 25 cm×25 cm，数量为 40 个，如图 6 b)所示。



a) 稀疏障碍仿真环境示意图　　b) 密集障碍仿真环境示意图

图 6　Webots 仿真实验环境示意图

Fig. 6　Schematic diagram of Webots simulation experiment environment

仿真环境中参数设置会影响其训练效果和稳定性，所以实验前需要初始化 CPG 网络参数和改进的 DDPG 算法参数，以及随机部署障碍物位置、目标点和蛇形机器人初始位置。DDPG 算法参数如表 1 所示，

CPG 网络参数如表 2 所示。

表 2 CPG 网络超参数设置

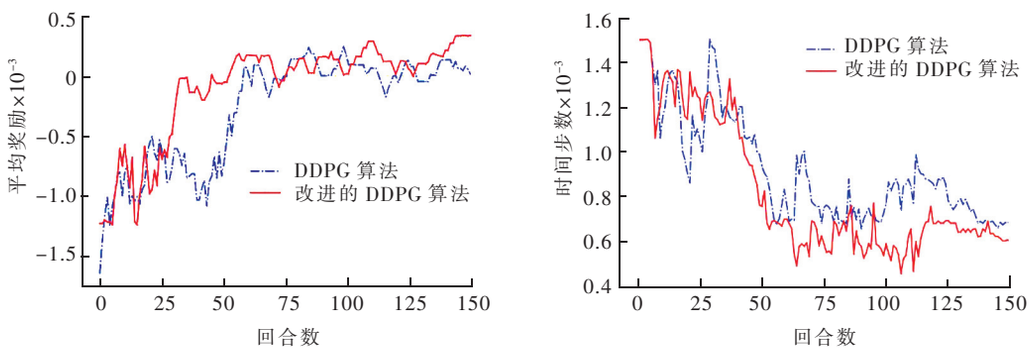
Tab. 2 CPG network hyperparameter settings

项 目	a	b	τ_r	τ_a	w_{ij}	w_{ji}
优化值	4.091 9	9.086 9	0.785 4	0.140 6	8.998 3	0.382 7

3.2 稀疏障碍物环境仿真

为了验证改进的 DDPG 算法的有效性，将其与原算法分别部署到稀疏障碍物地图中，通过设置奖励和相关约束，使蛇形机器人进行训练并积累经验，从而完成路径规划任务。

为降低初始阶段算法的随机性，解决蛇形机器人进入死区区域造成训练时间过长等问题，设置训练模型每回合最大迭代次数为 1 500，并记录蛇形机器人每回合训练后的平均奖励值和完成任务所需的迭代次数。为了突出蛇形机器人到达目标点时的奖励状态，设置完成任务后呈现正奖励状态，未完成任务或陷入死区区域后呈现负奖励状态，如图 7 a)所示。



a) DDPG 算法改进前后训练奖励值　　b) DDPG 算法改进前后搜索时间

图 7 DDPG 算法改进前后训练奖励值及搜索时间对比

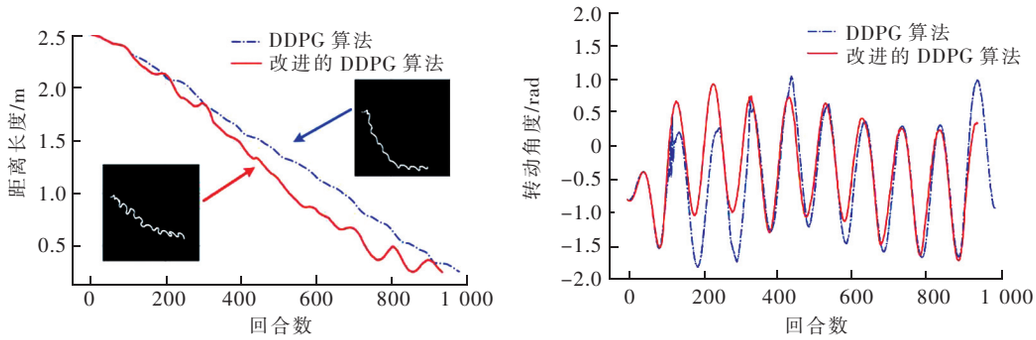
Fig. 7 Comparison of training reward value and search time before and after DDPG algorithm improvement

在稀疏障碍物环境下，改进 DDPG 算法的平均奖励值在回合数迭代 20 次以后开始出现明显的上升趋势，30 次后逐渐收敛，平均奖励值从负值逐渐趋近于 0，说明改进的 DDPG 算法已经积累了避障和趋近目标点的经验，随后由于算法积累经验不足和随机性，使蛇形机器人陷入死区区域，导致平均奖励值出现了短暂的下降趋势。由于 LSTM 算法控制样本信息的遗忘和记忆，平均奖励值短暂地下降之后，很快恢复至正值并且逐渐收敛，在算法训练回合迭代到 60 次以后，每次的平均奖励值在 0～250 之间波动且波动较小，如图 7 a)所示。

而 DDPG 算法训练回合迭代 40 次以后陷入死区区域，因其积累了大量劣质经验，所以需要较多的训练回合才能逐渐收敛，DDPG 算法训练回合迭代 60 次之后平均奖励值达到正值，80 次之后才逐渐收敛，收敛之后由于劣质经验的存在，导致蛇形机器人会陷入死区区域，从而造成目标任务无法完成。根据每回合最大迭代次数，搜索时间随着奖励值的收敛逐渐下降，改进的 DDPG 算法在训练回合迭代 120 次之后，每回合迭代次数逐渐稳定维持在 600～700 次，说明改进的 DDPG 算法具有在稀疏障碍物环境下完成路径规划任务的能力。DDPG 算法改进前后搜索时间如图 7 b)所示。

在稀疏地图环境下，分别测试 2 种算法训练好的模型。DDPG 算法完成蛇形机器人路径规划任务需要 987 次训练回合迭代，而改进的 DDPG 算法仅需要 938 次训练回合迭代。利用 Python 的 Pygame 功能包实时绘制其在地图中的任务轨迹，结果表明，改进的 DDPG 算法在完成任务时间和规划路径长度方面优于原算法，如图 8 a)所示。

另一方面，蛇形机器人节律运动的柔顺性直接影响了蛇形机器人的运动状态，其转弯动作主要依靠蛇头的偏转角度，在 CPG 网络控制下其余关节会随着蛇头角度变换进行跟随。由于障碍物的复杂性，改进 DDPG 算法使得蛇形机器人在路径规划任务中以较少的偏转次数到达目标点，而 DDPG 算法在执行路径规划任务时会出现短暂的连续转弯动作或者大幅度的转弯动作，从而造成蛇头曲线出现尖波和连续波动。根



a）DDPG 算法改进前后路径规划距离　　b）DDPG 算法改进前后头部关节角度变化

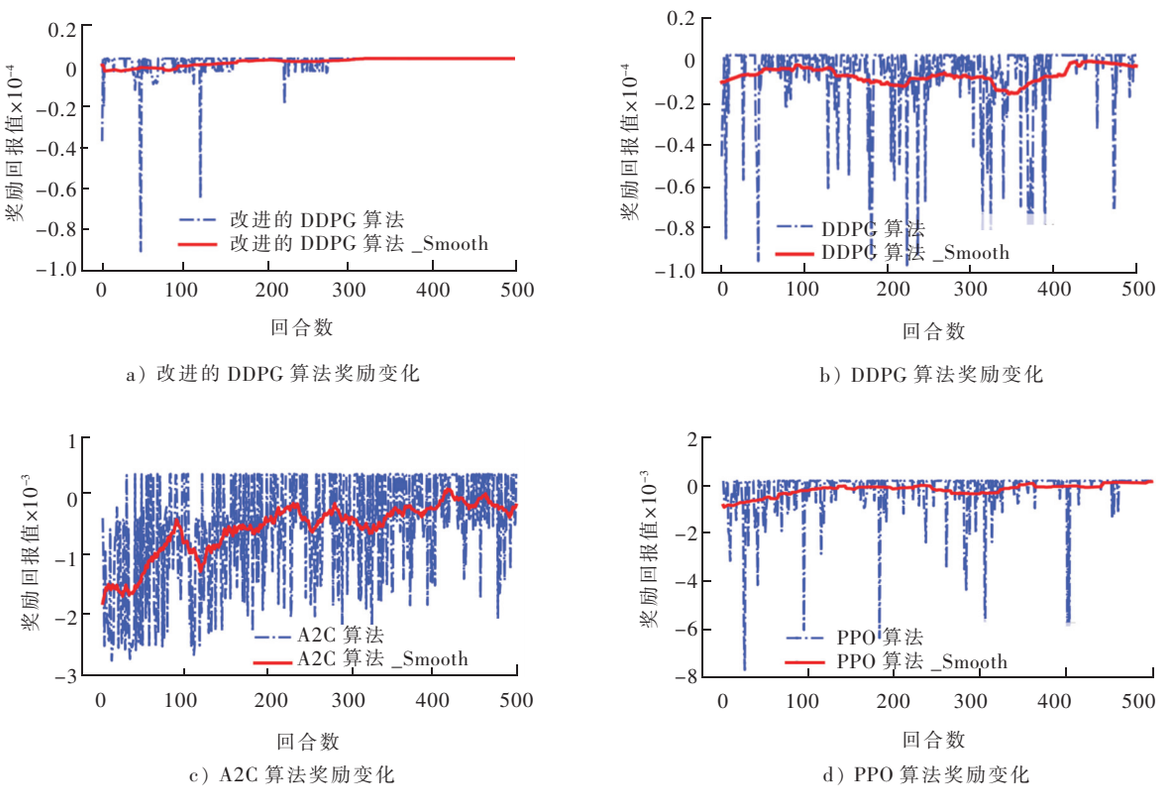
图 8　DDPG 算法改进前后行驶距离及头部角度变化对比

Fig. 8　Comparison of driving distance and head angle change before and after DDPG algorithm improvement

据仿真中蛇头关节角度变化，可以看到改进的 DDPG 算法规划路径相对于 DDPG 算法出现尖波和连续波动的情况较少，如图 8 b)所示。

3.3　密集障碍物环境仿真

在密集障碍物环境中，分别基于改进的 DDPG 算法、DDPG 算法、PPO 算法[21]和 A2C 算法[22]进行蛇形机器人路径规划任务，并对 4 种算法的迭代奖励值和搜索时间进行对比分析。改进的 DDPG 算法相对于其他算法，用更少的迭代次数积累了更多的优秀经验，奖励回报值和搜索时间曲线在训练回合迭代 200 次后逐渐收敛。面对死区问题借助多层 LSTM 神经网络的记忆功能可以更快速地适应环境，用尽量少的训练次数走出死区区域或重新规划路径，随着迭代次数的增加，蛇形机器人规划路径的长度逐渐减小并趋于稳定，训练回合迭代 300 次以后蛇形机器人能够更快地完成路径规划任务，如图 9、图 10 所示。

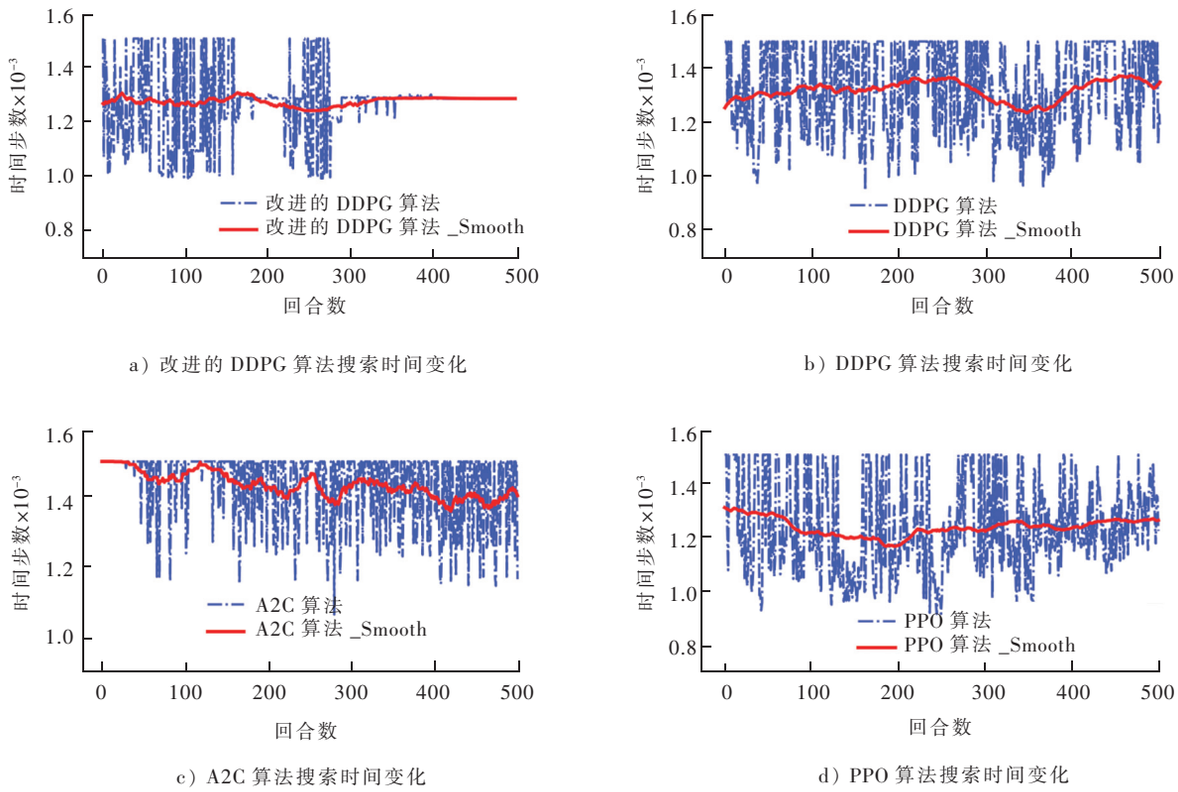


a）改进的 DDPG 算法奖励变化　　b）DDPG 算法奖励变化

c）A2C 算法奖励变化　　d）PPO 算法奖励变化

图 9　改进的 DDPG 算法、DDPG 算法、A2C 算法和 PPO 算法奖励对比

Fig. 9　Reward comparison of improved DDPG algorithm, DDPG algorithm, A2C algorithm and PPO algorithm



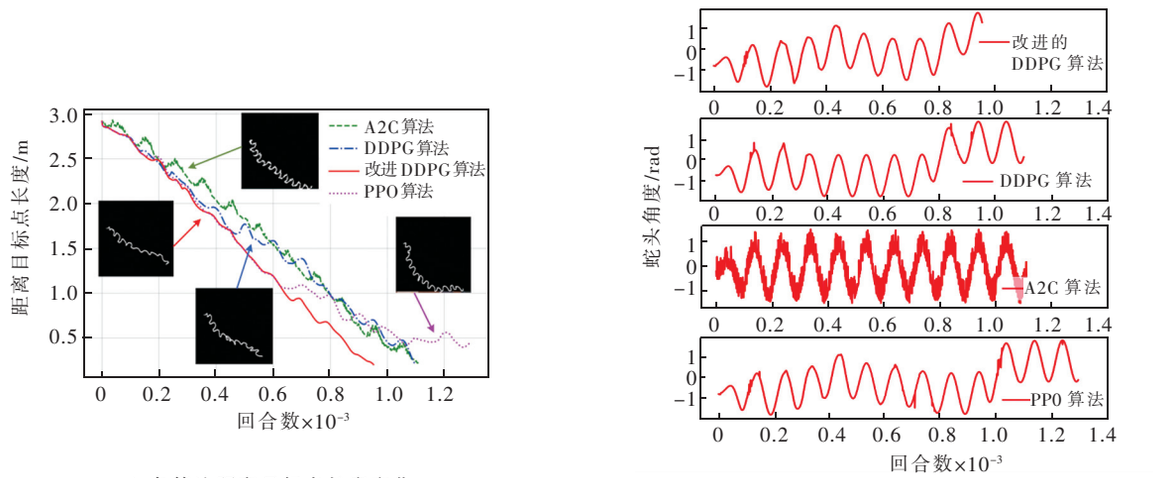
a）改进的 DDPG 算法搜索时间变化　　b）DDPG 算法搜索时间变化

c）A2C 算法搜索时间变化　　d）PPO 算法搜索时间变化

图 10　改进的 DDPG 算法、DDPG 算法、A2C 算法以及 PPO 算法搜索时间对比

Fig. 10　Search time comparison of improved DDPG algorithm, DDPG algorithm, A2C algorithm and PPO algorithm

相比于 DDPG 算法、PPO 算法和 A2C 算法，改进的 DDPG 算法在较短时间内找到了最优路径，其经过 978 次训练回合迭代后完成了路径规划任务且路径较为平滑。为了对比不同算法性能的优劣，在蛇形机器人执行路径规划任务中利用 Pygame 功能包生成规划路径曲线，通过仿真对比分析，改进的 DDPG 算法不仅规划的路径较为平滑且长度更短，而其他 3 种算法会陷入短暂的死区区域从而导致规划路径较长，如图 11 a)所示，密集障碍物环境下各算法路径规划长度对比如表 3 所示。通过对比分析不同算法下蛇形机器人的蛇头角度变化曲线可知，改进的 DDPG 算法不仅转弯角度更加平滑，而且没有连续的角度突变，如图 11 b)所示。



a）各算法距离目标点长度变化　　b）测试模型中各算法蛇头角度变化

图 11　改进 DDPG 算法、DDPG 算法、A2C 算法和 PPO 算法的终点距离和蛇头角度变化对比

Fig. 11　Improved DDPG algorithm, DDPG algorithm, A2C algorithm and PPO algorithm were improved to compare the change of end distance and snake head angle

面对连续避障情况时，因为状态空间和奖励值的设置，蛇形机器人会受到多种限制，出现短暂的决策过程。为了更加清晰地对比改进 DDPG 算法的优化效果，避免随机性结果的产生，每种算法使用相同的参数设置，实验对比了密集障碍物场景下 4 种算法的最优路径规划次数、陷入死区的次数、平均训练时间和模型测试中得到的平均规划路径长度，分别进行 10 次训练并取其平均值，如表 4 所示。

表 3　密集障碍物环境下算法路径长度对比表

Tab. 3　Algorithm path length comparison table in dense obstacle environment

算　法	规划路径长度/m
改进的 DDPG 算法	7.869
DDPG 算法	9.557
PPO 算法	12.724
A2C 算法	13.890

表 4　模型训练参数设定

Tab. 4　Model Training Parameter Setting

指　　标	改进的 DDPG 算法	DDPG 算法	PPO 算法	A2C 算法
最优路径规划次数	421	285	376	186
陷入死区次数	17	92	47	137
平均训练时间/min	496	577	523	611
平均规划路径迭代次数	954	1 102	1 112	1 296

从表 4 可以看出，改进的 DDPG 算法在训练中可以更快地收敛，其整体训练时间相比于其他算法平均降低了 15%，模型测试中完成规划路径迭代次数降低了 22%，提高了算法的快速性。为使蛇形机器人更好地适应环境以完成路径规划任务，改进的 DDPG 算法因为多层 LSTM 神经网络的选择记忆功能，每次训练只会在训练前期陷入死区区域，并且通过更多优质经验的积累提高了网络模型训练的稳定性。

4　结　语

本文提出的改进 DDPG 算法，可以有效解决蛇形机器人在选择最优路径时的局部死区问题和路径规划算法训练速度慢的问题。通过改变网络结构、引入 LSTM 神经网络模型，快速积累高奖励值样本，获得优质经验；同时，对搭建的 CPG 网络模型进行最优化特征参数处理，并将其融入 DDPG 算法中，结合新型状态空间和奖励函数，进而更好地引导蛇形机器人完成路径规划任务。仿真结果表明，改进算法具有收敛速度快、训练时间短、完成任务迭代次数多和陷入死区次数少等优势。因此本文提出的算法可使蛇形机器人能够在复杂环境中自主进行路径规划，为实际应用中更加安全且快速地完成导航与控制任务提供了有价值的解决方案。

本文主要研究了蛇形机器人平面路径规划，但未考虑地形起伏变化。未来研究拟基于蛇形机器人的多模式运动，改进运动控制策略和路径规划方法，提高机器人的环境适应能力。

参考文献/References：

[1] PETTERSEN K Y. Snake robots[J]. Annual Reviews in Control, 2017, 44: 19-44.

[2] LIU Jindong, TONG Yuchuang, LIU Jinguo. Review of snake robots in constrained environments[J]. Robotics and Autonomous Systems, 2021, 141. DOI: 10.1016/j.robot.2021.103785.

[3] YU Xue, CHEN Weineng, GU Tianlong, et al. ACO-A*: Ant colony optimization plus A* for 3-D traveling in environments with dense obstacles[J]. IEEE Transactions on Evolutionary Computation, 2019, 23(4): 617-631.

[4] HAN Siwei, XIAO Wenyu, YU Zhenghong, et al. Adaptive climbing gait design of snake robot based on extended Kalman filter[J]. Journal of Physics: Conference Series, 2022. DOI: 10.1088/1742-6596/2183/1/012003.

[5] BORHAUG E, PAVLOV A, PETTERSEN K Y. Integral LOS control for path following of underactuated marine surface vessels in the presence of constant ocean currents[C]//2008 47th IEEE Conference on Decision and Control. Cancun: IEEE, 2008: 4984-4991.

[6] 郑莹，段庆洋，林利祥，等. 深度强化学习在典型网络系统中的应用综述[J]. 无线电通信技术，2020，46(6)：603-623.
ZHENG Ying, DUAN Qingyang, LIN Lixiang, et al. A survey on the applications of deep reinforcement learning in classical networking systems[J]. Radio Communications Technology, 2020, 46(6): 603-623.

[7] JIA Yuanyuan, MA Shugen. A coach-based Bayesian reinforcement learning method for snake robot control[J]. IEEE Robotics and Automation Letters, 2021, 6(2): 2319-2326.

[8] BING Zhenshan, LEMKE C, CHENG Long, et al. Energy-efficient and damage-recovery slithering gait design for a snake-like robot based on reinforcement learning and inverse reinforcement learning[J]. Neural Networks, 2020, 129: 323-333.

[9] LIU Xuan, GASOTO R, JIANG Ziyi, et al. Learning to locomote with artificial neural-network and CPG-based control in a soft snake robot [C]//2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Las Vegas: IEEE, 2020: 7758-7765.

[10] 严浙平，杨皓宇，张伟，等. 基于模型预测-中枢模式发生器的六足机器人轨迹跟踪控制[J]. 机器人，2023，45(1)：58-69.
YAN Zheping, YANG Haoyu, ZHANG Wei, et al. Trajectory tracking control of hexapod robot based on model prediction and central pattern generator[J]. Robot, 2023, 45(1): 58-69.

[11] QIN Guodong, WU Huapeng, CHENG Yong, et al. Adaptive trajectory control of an under-actuated snake robot[J]. Applied Mathematical Modelling, 2022, 106: 756-769.

[12] JIANG Zhuangyi, OTTO R, BING Zhenshan, et al. Target tracking control of a wheel-less snake robot based on a supervised multi-layered SNN[C]//2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Las Vegas: IEEE, 2020: 7124-7130.

[13] 张瀚，解明扬，张民，等. 融合 DDPG 算法的移动机器人路径规划研究[J]. 控制工程，2021，28(11)：2136-2142.
ZHANG Han, XIE Mingyang, ZHANG Min, et al. Path planning of mobile robot with fusion DDPG algorithm[J]. Control Engineering of China, 2021, 28(11): 2136-2142.

[14] CHO Y, MANZOOR S, CHOI Y. Adaptation to environmental change using reinforcement learning for robotic salamander[J]. Intelligent Service Robotics, 2019, 12(3): 209-218.

[15] CARRARA F, FALCHI F, CALDELLI R, et al. Detecting adversarial example attacks to deep neural networks[C]//Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing. Florence: Association for Computing Machinery, 2017: 1-7.

[16] PENG Xuebin, BERSETH G, van de PANNE M. Terrain-adaptive locomotion skills using deep reinforcement learning[J]. ACM Transactions on Graphics, 2016, 35(4): 1-12.

[17] PENG Xuebin, BERSETH G, YIN Kangkang, et al. DeepLoco: Dynamic locomotion skills using hierarchical deep reinforcement learning [J]. ACM Transactions on Graphics, 2017, 36(4): 1-13.

[18] CHEN Chewen, TSENG S P, KUAN Tawen, et al. Outpatient text classification using attention-based bidirectional LSTM for robot-assisted servicing in hospital[J]. Information, 2020, 11(2). DOI: 10.3390/info11020106.

[19] YU Yong, SI Xiaosheng, HU Changhua, et al. A review of recurrent neural networks: LSTM cells and network architectures[J]. Neural Computation, 2019, 31(7): 1235-1270.

[20] LIAO Xiaocun, ZHOU Chao, ZOU Qianqian, et al. Dynamic modeling and performance analysis for a wire-driven elastic robotic fish[J]. IEEE Robotics and Automation Letters, 2022, 7(4): 11174-11181.

[21] YANG Laiyi, BI Jing, YUAN Haitao. Dynamic path planning for mobile robots with deep reinforcement learning[J]. IFAC-PapersOnLine, 2022, 55(11): 19-24.

[22] XING Xiangrui, DING Hongwei, LIANG Zhuguan, et al. Robot path planner based on deep reinforcement learning and the seeker optimization algorithm[J]. Mechatronics, 2022, 88. DOI: 10.1016/j.mechatronics.2022.102918.