doi: 10.7690/bgzdh.2022.06.008

降低方差的深度确定性策略梯度算法

赵国庆，徐君明，刘爱东
（海军航空大学岸防兵学院，山东 烟台 246001）

摘要：针对高方差现象导致训练过程不稳定、算法性能下降的问题，提出一种降低方差的深度确定性策略梯度算法（reduction variance deep deterministic policy gradient，RV-DDPG）。通过延迟更新目标策略的方法，减少误差出现次数，降低误差的累计；通过平滑目标策略的方法，减小单步误差，稳定方差。将 RV-DDPG 算法、传统深度确定性策略梯度算法（deep deterministic policy gradient，DDPG）和目前广泛应用的异步优势行动者评论家算法（asynchronous advantage actor-critic，A3C）应用于 Pendulum、Mountain Car Continues 和 Half Cheetah 问题。实验结果表明：RV-DDPG 具有更好的收敛性和稳定性，证明了该算法降低方差的有效性。

关键词：强化学习；DDPG；平滑目标策略；策略延迟更新；降低方差
中图分类号：TJ02　　**文献标志码：**J

Deep Deterministic Policy Gradient Algorithm with Reduced Variance

Zhao Guoqing, Xu Junming, Liu Aidong
(*School of Coastal Defence*, *Naval Aviation University*, *Yantai* 246001, *China*)

Abstract: In order to solve the problem that high variance leads to the instability of training process and the decline of algorithm performance, a reduction variance deep deterministic policy gradient (RV-DDPG) algorithm is proposed. Through the method of delaying updating the target strategy, the number of errors is reduced and the accumulation of errors is reduced; through the method of smoothing the target strategy, the single-step error is reduced and the variance is stabilized. The RV-DDPG algorithm, the traditional deep deterministic policy gradient algorithm (DDPG) and the widely used asynchronous asynchronous advantage actor (A3C) are applied to Pendulum, Mountain Car Continues and Half Cheetah problems. The experimental results show that RV-DDPG has better convergence and stability, which proves the effectiveness of the algorithm to reduce the variance.

Keywords: reinforcement learning; DDPG; smooth target strategy; policy delay update; reduction variance

0　引言

强化学习[1]是机器学习中最接近大众所设想的人工智能模样的领域，假设智能体（Agent）通过在所处任务环境中不断尝试并吸取经验，最后成长为解决相关问题的专家，主要用于解决智能体序贯决策问题。简言之，强化学习希望智能体在一个环境中，随着“时间的流逝”，不断地自我学习，并最终在这个环境中学到最为合理的行为策略，即一系列针对不同情形的最合理的行为组合逻辑。

目前强化学习在机器人控制、计算机视觉、参数优化、博弈论和组合优化调度等领域得到了广泛应用[2]，为经济社会发展作出了重要贡献。比如：Ugurlu 等[3]采用双延迟算法研究经济领域的预测问题；AL-MARRIDI 等[4]对连续动作的强化学习方法进行了研究，并将其应用到机器人避障行为中；陈红名[5]针对深度确定性策略梯度（DDPG）探索方式的不稳定，提出了基于经验指导的方法；吴俊塔[6]针对 DDPG 算法训练不稳定、时间长的问题，提出了用于集成的多深度确定性策略梯度算法，并通过仿真实验证明了其有效性。

在强化学习领域，研究热点为以 Actor-Critic 框架[7]为基础的各类算法，比如异步优势行动者评论家（A3C）[8]、深度确定性策略梯度算法[9]、双延迟深度确定性策略（twin delayed deterministic policy gradient，TD3）[10]等。笔者以 DDPG 算法为研究对象，针对其因误差累积出现的过拟合现象，提出基于降低方差的深度确定性策略梯度算法（RV-DDPG），并在实验中进行了对比验证，RV-DDPG 算法波动性更小，且具有更好的收敛特性。

1　DDPG 算法结构与问题分析

1.1　强化学习基础理论

在强化学习问题中，智能体要和环境完成一系

收稿日期：2022-02-15；**修回日期：**2022-03-28
基金项目：2020 海军军事理论研究课题
作者简介：赵国庆（1991—），男，山东人，硕士，从事反无人机机器学习、强化学习、任务规划研究。E-mail: 124116846@qq.com。

列交互，在每一时刻，环境处于一种状态 s_t，智能体根据当前状态的观测值做出行动 a_t 之后，环境会给出反馈，即奖励 r_{t+1} 和新的状态 s_{t+1}。循环即可不断地得到一个短序列：$s_t \to a_t \to r_t \to s_{t+1}$。

在强化学习中，智能体与环境的交互构成了一个马尔可夫决策过程(markov decision process，MDP)[11]，用四元组 $\langle s,a,r,p\rangle$ 表示。其中，概率 p 表示前一个状态 s_t 转移到下一个状态的概率 s_{t+1} 的概率，假定其满足条件 $p[s_{t+1}|s_t]=p[s_{t+1}|s_1,\cdots,s_t]$，即系统的下一状态 s_{t+1} 仅与当前状态 s_t 有关，而与以前的状态无关，称其为马尔可夫性。

强化学习的目标是给定一个马尔可夫决策过程，寻找最优策略 π，即一系列状态到动作的映射，使得总回报最大。当智能体采用策略 π 时，累积回报在状态 s 处的期望值是一个确定值，定义状态-行为值函数：

$$q_\pi(s,a)=E_\pi\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}\,|\,s_t=s,a_t=a\right]。\qquad(1)$$

经变换，可得到贝尔曼方程[12]，任意一个状态的价值可通过递归的形式由其他状态的价值得到：

$$q_\pi(s,a)=E_\pi[r_{t+1}+\gamma q_\pi(s_{t+1},a_{t+1})\,|\,s,a]。\qquad(2)$$

计算状态值函数的目的是为了构建学习算法从数据中得到最优策略。每个策略对应一个状态值函数，最优策略自然对应最优状态值函数。状态-行为值函数的贝尔曼最优方程：

$$q_\pi^*(s,a)=R_s^a+\gamma\sum_{s'\in s}p_{ss'}^a\max_{a'}q^*(s',a')。\qquad(3)$$

由此，最大化 $q_\pi^*(s,a)$ 可得最优策略：$\pi_*(a|s)=\arg\max_{a\in A}q_*(s,a)$。

1.2 DDPG 算法结构

DeepMind 科研团队利用 DQN[13]扩展 Q 学习算法的方式，通过利用深度神经网络对状态-行为值函数 $q^\omega(s,a)$ 和确定性策略 $\mu_\theta(s)$进行逼近。DDPG 算法引入神经网络结合 DQN 算法的原理，很好地解决了值函数收敛问题，在高维连续动作空间问题上表现出色，而且使值函数收敛问题也得到了解决。DDPG 借鉴了 DQN 的 2 个技巧：经验回放和目标网络，解决了数据独立同分布的需求，提升了算法的稳定性，网络的训练过程大大提升。DDPG 算法框架如图 1 所示。

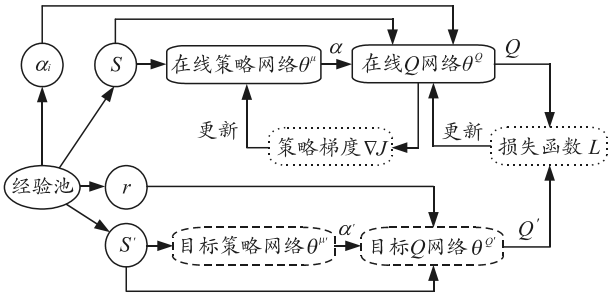


图 1 DDPG 算法框架

最基本的 2 个参数为 μ 和 Q。μ 为确定性策略，每一步的行为都用策略函数 μ 获得最大值，即 $a_t=\mu(s_t|\theta^\mu)$；Q 函数，即行为-值函数，表示在状态 s_t 下，采取动作 a_t 后，且如果持续执行策略的情况下，所获得的 r_t 期望值：θ^Q

$$Q^\mu(s_t,a_t)=E[r(s_t,a_t)+\gamma Q^\mu(s_{t+1},a_{t+1})]。\qquad(4)$$

在当前状态空间中根据 μ 选择动作能够取得的动作状态值的和，称为函数 J，以此衡量策略 μ 的表现：

$$J_\beta(\mu)=\int_S\rho^\beta(s)Q^\mu(s,\mu(s))ds=E_{s\sim\rho^\beta}[Q^\mu(s,\mu(s)]。\qquad(5)$$

式中：s 为基于策略 μ 产生的状态序列；分布函数为 ρ^β，$Q^\mu(s,\mu(s))$为在每个状态下，按照确定性策略产生的 Q 值；因此 $J_\beta(\mu)$可理解为状态 s 根据 ρ^β 分布时，$Q^\mu(s,\mu(s))$的期望值，训练目标即最大化 $J_\beta(\mu)$，即 $\mu=\arg\max_\mu J(\mu)$。

DDPG 采取在线(online)网络和目标(target)网络相结合的方式稳定训练过程，构建了 4 个基本网络：

1) 在线策略网络：用一个卷积神经网络对 μ 进行模拟，其参数为 θ^μ，负责根据当前状态 s_t 选择当前动作 a_t，用于和环境交互生成 r_t 和 s_{t+1}，训练 μ 网络的过程，就是寻找 μ 网络参数 θ^μ 最优解的过程。

2) 在线 Q 网络：用一个卷积神经网络对 Q 函数进行模拟，其参数为 θ^Q，负责计算当前 Q 值，训练 Q 网络的过程，就是寻找 Q 网络参数 θ^Q 最优解的过程。

3) 目标策略网络：将在线策略网络的神经网络拷贝得到该网络，其参数为 $\theta'^{\mu'}$，网络参数定期从 μ 复制，负责根据经验回放池中采样的下一状态 s_{t+1}，选择最优下一动作 a_{t+1}。

4) 目标 Q 网络：将在线目标网络的神经网络拷贝得到该网络，其参数为 $\theta^{Q'}$，负责计算目标 Q 值中的 Q' 部分，网络参数定期从 Q 复制。其中目标 Q 值为 $y_i=r_i+\gamma Q'(s_{i+1},\mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$。

1.3　问题分析

DDPG 算法中使用神经网络拟合了值函数，其函数评估模型的输出值含有估计误差，估计值不能确切反映真实值。算法在当前状态下选择动作时采用的是 max 操作，模型总是会倾向于被放大的状态，造成高方差问题。

假设函数评估模型在状态 s 下采取动作 a 所带来的估计值为 $q_{\text{appro}}(s,a)$，目标值为 $q_{\text{true}}(s,a)$，函数评估模型所带来的评估误差由 δ_s^a 表示，假设误差为服从均匀分布的随机误差，即 $\delta_s^a \sim U[-\varepsilon,\varepsilon]$，$\varepsilon$ 为其上限，如下式：

$$\mathrm{q}_{\text{appro}}(s,a)=\mathrm{q}_{\text{true}}(s,a)+\delta_s^a \text{。} \tag{6}$$

估计值与真实值在状态 s 下的误差为：

$$\begin{aligned} b_s =& (r_s^a+\gamma\max_{a'} q_{\text{appro}}(s',a')- \\ & (r_s^a+\gamma\max_{a'} q_{\text{true}}(s',a'))= \\ & \gamma(\max_{a'} q_{\text{appro}}(s',a')-\max_{a'} q_{\text{true}}(s',a'))= \\ & \gamma(\max_{a'}(q_{\text{true}}(s',a')+\delta_{s'}^{a'})- \\ & \max_{a'} q_{\text{true}}(s',a'))=\gamma(\max_{a'}\delta_{s'}^{a'}) \text{。} \end{aligned} \tag{7}$$

由 $E[\delta_{s'}^{a'}]=0$，得 $P(\max_a \delta_s^a>0)>P(\max_a \delta_s^a<0)$。

综上，$\forall a \to E[b_s]>0$，$q_{\text{appro}}(s,\ a)$大概率大于 $q_{\text{true}}(s,a)$，单次更新的估计值往往大于真实值。

DDPG 算法为时序更新的，使用后续状态的估计值估计当前状态的估计值，虽然单步更新的误差比较小，但是训练时这些估计误差会在贝尔曼方程得到累积或者恶化。

根据贝尔曼方程，每次执行单步更新中都会存在误差 δ_t，称之为 TD-error：

$$q_\theta(s_t,a_t)=r_t+\gamma E[q_\theta(s_{t+1},a_{t+1})]-\delta_t \text{。} \tag{8}$$

将上式展开，得：

$$\begin{aligned} & q_\theta(s_t,a_t)=r_t+\gamma E[q_\theta(s_{t+1},a_{t+1})]-\delta_t= \\ & r_t+\gamma E[r_{t+1}+\gamma E[q_\theta(s_{t+2},a_{t+2})]-\delta_{t+1}]-\delta_t= \\ & E_{s_i\sim\rho_\pi,a_i\sim\pi}[\sum_{i=T}^{T}\gamma^{i-t}(r_i-\delta_i)] \text{。} \end{aligned} \tag{9}$$

Q 估计值是期望的返回值减去期望 TD-error 的和，即 Q 估计值的方差是同实时奖励与误差的方差成正相关的，如果不控制单步更新的误差 δ_t，Q 估计值中的累积误差 $\sum_t \delta_t$ 会在整个回合中传播，导致 Q 值的方差增大，进而影响准确性。

2　RV-DDPG 算法设计

针对高方差问题，对算法进行了改进，提出了降低方差的深度确定性策略梯度算法 RV-DDPG，主要做了 2 点优化：1) 通过策略延迟，减少误差出现次数，降低误差的累计；2) 通过策略平滑，减小单步误差，稳定方差。

2.1　策略延迟更新

传统的 DDPG 算法中，策略网络和 Q 网络在连续状态中参数更新，存在一定相关性，由于 Q 网络估值的误差会产生次优策略，并在策略网络的参数更新中得到加强，朝着错误的方向更新，并用此训练 Q 网络，最终导致 2 个网络的劣化循环。

Deep Mind 在 DQN 算法中指出，目标网络可以用于减少多步更新的误差，降低策略在高方差状态下发散的可能性，提升强化学习的稳定性。并且由于计算网络目标值需要用到现有的 Q 值，因此笔者采用一个更新较慢的策略网络专门提供此 Q 值，原网络的 Q 值仅用于动作选择和更新参数，即为策略延迟更新。

通过此方法，一方面减少不必要的重复更新，另一方面也减少在多次更新中累积的误差，在一定程度上缓解了高方差问题，提高了训练的稳定性和收敛性。

RV-DDPG 算法中，取策略延迟的长度为 2，即 Q 函数每更新 2 次后再进行策略的更新。同时，为了使误差保持较小水平，S-DDPG 对于 2 个目标网络同样使用软更新的方式，使参数逐渐逼近，大幅增强学习的稳定性和收敛能力：$\theta' \leftarrow \tau\theta+(1-\tau)\theta'$。

2.2　目标策略平滑

根据上文分析，由于误差的传播和累积，在一个局部范围内某个动作的 Q 估计值可能特别大，发生特殊的故障模式。如果 Q 函数为此动作学到了一个错误的峰值，策略将快速利用该峰值，进而出现脆弱或者不正确的动作。在强化学习领域常用的解决方法是对参数更新进行正则化，笔者采用策略平滑的方法，通过在目标策略网络中引入噪声，对相似动作的每一个维度进行平滑，减少误差的产生，进而降低方差，使值函数更加平滑。

Ornstein-Uhlenbeck（OU）随机过程作为噪声，其微分方程为：

$$dx_t=\theta(\mu-x_t)dt+\sigma dW_t \text{。} \tag{10}$$

参数 θ=0.15、均值 μ=0、方差 σ=1 的噪声图像如图 2 所示。

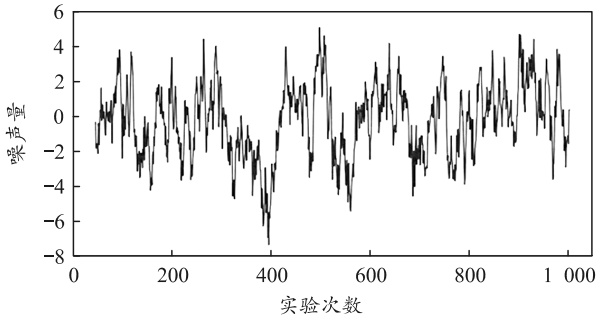


图 2 OU 随机过程

同时，参考近端策略优化算法(proximal policy optimization，PPO)中保守策略迭代的思路，在噪声中加入 clip 裁剪函数，将输出的最大值和最小值进行了限定，防止出现较大的变化。综上，目标函数为：

$$y = r + \gamma q_{\theta'}(s', \pi_{\phi'}(s') + \varepsilon)$$

$$\varepsilon \sim \text{clip}(N(0,\sigma), -c, c) \quad 。 \tag{11}$$

通过对目标策略进行平滑处理，避免选取某些异常的峰值而出现错误，降低了方差，提升了策略更新速度和网络的稳定性，加速了学习。

2.3 RV-DDPG 算法

笔者提出的 RV-DDPG 算法流程如下：

随机初始化在线网络参数 θ^{μ} 和 θ^{Q}

将在线网络参数分别复制给对应的目标网络 $\theta'^{\mu'}$ 、 $\theta^{Q'}$

初始化经验回放池 D

For each episode：

初始化 UO 随机过程

For t=1,T:

actor 在状态 s 下基于当前策略得到动作 $a_t = \mu(s_t \mid \theta^{\mu}) + \varepsilon$ ， $\varepsilon \sim N(0,\sigma)$

执行动作 a_t，得到新的状态 s_{t+1}，奖励 r_t

将四元组 $\{s_t, a_t, r_t, s_{t+1}\}$ 存入经验回放集 D，作为训练在线网络的数据集

从经验回放集合 D 中采样 N 个样本 $\{s_j, a_j, r_j, s_{j+1}\}$，作为在线策略网络和在线 Q 网络的一个 mini-batch 训练数据

计算在线 Q 网络的梯度：

使用均方差损失函数

$$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i \mid \theta^{Q}))^2 ,$$

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} \mid \theta^{\mu'}) \mid \theta^{Q'} + \varepsilon) ,$$

$$\varepsilon \sim \text{clip}(N(0,\sigma), -c, c) 。$$

通过 Adam 优化器更新在线 Q 网络的参数 θ^{Q}

If t mod policy_delay = 0:

计算策略网络的梯度：

$$\nabla_{\theta^{\mu}} J_{\beta}(\mu) = \frac{1}{N}\sum_i \nabla_a Q(s, a \mid \theta^{Q})|_{s_i, \mu(s_i)} \nabla_{\theta^{\mu}} \mu(s \mid \theta^{\mu})|_{s_i})$$

通过 Adam 优化器更新在线策略网络参数 θ^{μ}

使用软更新对 2 个目标网络进行更新：

$$\left.\begin{aligned} \theta^{Q'} &\leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} &\leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \\ \tau &<< 1 \end{aligned}\right\}$$

End for time step

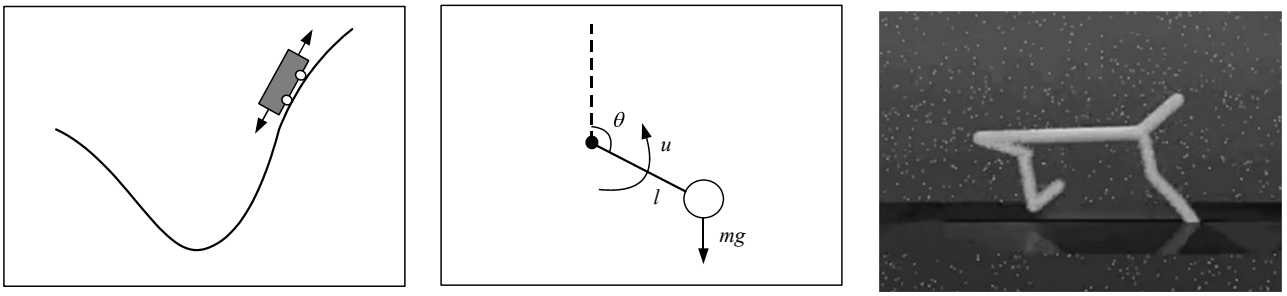
End for episode

3 算法仿真与结果分析

采用 OpenAI GYM[14]平台上 classic control 和 Mujoco 物理模拟器[15]中的环境进行实验。使用 3 个连续环境来对 RV-DDPG 算法进行测评，同时与传统 DDPG 算法和 A3C 算法进行了对比。所使用 3 个环境包括 Pendulum、Mountain Car Continues 和 Humanoid。

3.1 实验描述

几个连续任务状态空间和动作空间设计思路如图 3 所示。



(a) Mountain Car Continues (b) Pendulum (c) Humanoid

图 3 实验环境

1) Mountain Car Continues 问题的主体是一个可以左右驱动的小车，学习目标是确定油门操作策略，使小车能以尽量短的时间从山谷最低点到达右侧山顶。小车上山问题中，小车的位置、速度为状态变量，小车油门实施的操作为动作变量，取值范围分别为 $x_t \in [-1.5, 1.5]$ 、$x_t' \in [-0.07, 0.07]$ 、$a_t \in [-1, 1]$；奖励函数设计如下：初始奖励为 0，当小车成功到达右侧山顶的时候，r=100；当小车在循环结束时仍未达到目标，$r_{t+1} = r_t - 0.1a^2$。

2) Pendulum 问题主体是一个摆杆，学习目标是确定力矩控制策略，以尽可能迅速地将摆杆从随机起始位置举起并长时间保持平衡；倒立摆起摆问题中，摆杆偏离垂直方向的角度与角速度组成 2 维状态变量，目标是保持垂直，即旋转速度最小，力度最小，对摆杆绕旋转轴施加的力矩为 1 维动作变量，取值范围分别为 $\theta_t \in [-\pi, \pi]$ 、$\theta_t' \in [-78.54, 78.54]$ 、$\mu_t \in [-2, 2]$，奖励函数：

$$R = -(\theta^2 + 0.1 \times \theta'^2 + 0.001 \times \text{action}^2)。\tag{12}$$

3) Half Cheetah 问题主体是一个双足机器人，学习目标是奔跑的步态，使其在 2 维空间快速奔跑。状态空间为 17 维，由角色位置、关节动作、速度等信息组成，动作空间为 6 维，由各关节的扭矩控制组成，控制各关节的速度。奖励函数 R=−(0.1×action^2)+(X_{after}−X_{before})，采取的是定型的思路，考虑控制效果和位置变化 2 个因素，即机器人越接近奖励目标，系统给予的奖励就越多。

在强化学习问题中，智能体可以通过此接口获取观测的状态，然后根据策略选择相应的动作并再次同环境交互，环境获取交互后，提供新的观测和奖励，直至任务结束。在不断地探索、试错过程中，寻找到最大奖励期望。

3.2　参数设置

对于 RV−DDPG，主要涉及训练中的经验回放所使用的内存大小、折扣率、学习率、网络参数更新率等。参数设置如表 1 所示。

表 1　超参数设置

变量名	变量含义	数值大小
LR	学习率	0.001
GAMMA	奖励的折扣	0.920
TAU	软更新	0.010
MEMORY_CAPACITY	经验池大小	10 000
POLICY_DELAY	策略延迟	2.000
BATCH_SIZE	批尺寸	32.000

实验中使用的原始 DDPG 算法和 A3C 算法参考 OpenAI Baselines 的强化学习算法集合。

3.3　测试结果与分析

由于强化学习训练的目标是最大化奖励，所以借助累积回报曲线对算法的性能和智能体训练效果进行评价，从达到收敛的时间和收敛后的波动性 2 方面进行分析。累积回报是智能体在做出相应动作后，环境反馈的累积奖励，通过程序将智能体训练回合累积回报输出如图 4 所示。

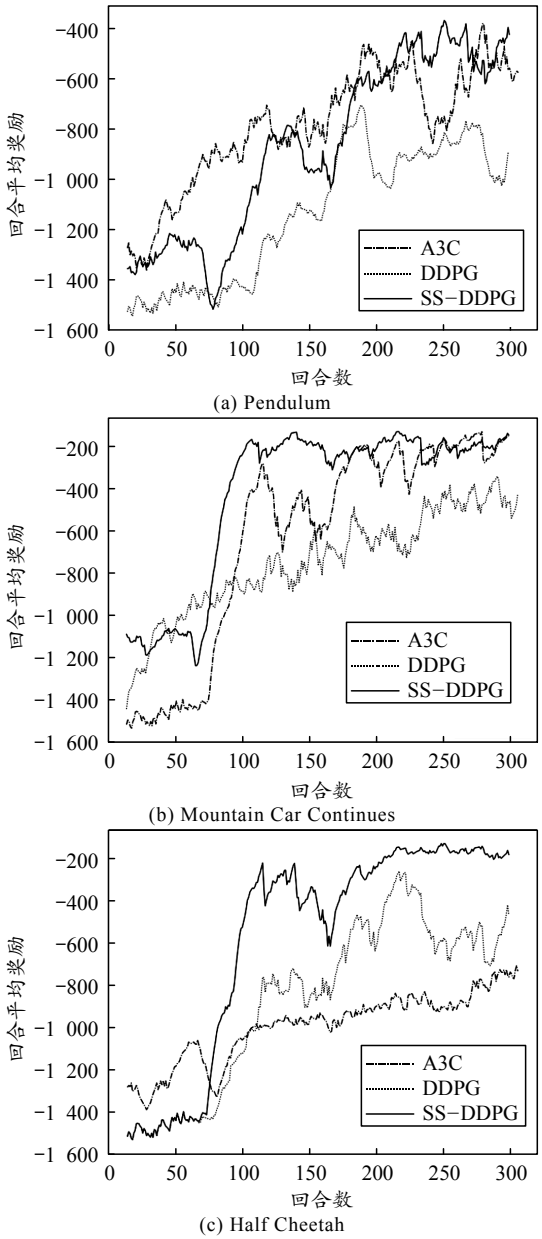


图 4　仿真实验结果

在 Pendulum 中，3 种算法均呈现稳定的上升趋势。其中，A3C 算法在 100 回合左右到达平均累积回报，提升速度较其他 2 种算法更快。改进的

RV-DDPG 算法，在 130 回合处出现较大波动，累积回报有较大幅度下降，在 180 回合之后呈现较为稳定的状态，并且平均累积回报与 A3C 算法相近，均比原始 DDPG 算法要高很多。而原始的 DDPG 算法，在达到收敛的时间和收敛平均值方面，较其他 2 种算法差距较大。

在 Mountain Car Continues 中，RV-DDPG 算法在 100 回合处到达平均累积回报，之后一直保持较好的稳定性，未出现较大波动，在此实验中性能最为出色。A3C 算法同样在 100 回合到达平均累积回报，但随后出现较大波动，直到 170 回合处才达到较为稳定的状态。原始 DDPG 算法中，平均累积回报自起始处于稳定的增长状态，在 250 回合处到达最大值，呈现稳定状态，但平均累积回报较其他 2 种算法相比有一定差距。

在 Half Cheetah 中，A3C 算法平均累积回报增长缓慢，且平均值较其他 2 种算法低很多。原始 DDPG 算法在 300 回合处依然呈现较大波动，未达到收敛状态。RV-DDPG 算法在 180 回合处达到稳定的平均累积回报，实现了稳定的收敛。

通过以上对比发现，在学习速度上，RV-DDPG 由于采用了策略延迟更新，优化解决了 Actor 和 Critic 的耦合性问题，进而可以更快达到全局最优解。在对环境的探索上，由于对目标网络的策略进行了平滑处理，避免了一些极端态，所以方差更小，学习更加稳定。

4 结论

针对现有 DDPG 算法在求解过程中方差过高的问题展开研究，提出降低方差的深度确定性策略梯度算法。该算法通过策略延迟更新和目标策略平滑 2 种方法，降低了算法的方差。通过仿真实验对改进的算法进行了对比验证，结果表明：RV-DDPG 算法的学习性能和效率均优于原始 DDPG 算法。

参考文献：

[1] 汤佳欣, 陈阳, 周孟莹. 深度学习方法在兴趣点推荐中的应用研究综述[J/OL]. 计算机工程: 1-15[2021-08-16]. https//doi. org/10. 19678/j. issn. 1000-3428. 0061598.

[2] 张荣霞, 武长旭, 孙同超, 等. 深度强化学习及在路径规划中的研究进展[J/OL]. 计算机工程与应用: 1-15[2021-08-16]. http//kns. cnki. net/kcms/detail/11. 2127. tp. 20210714. 1447. 008. html.

[3] UGURLU H I, KALKAN S, SARANLI A. Reinforcement Learning versus Conventional Control for Controlling a Planar Bi-rotor Platform with Tail Appendage[J]. Journal of Intelligent & Robotic Systems, 2021, 102(4).

[4] AL-MARRIDI A Z, MOHAMED A, ERBAD A. Reinforcement learning approaches for efficient and secure blockchain-powered smart health systems[J]. Computer Networks, 2021(prepublish).

[5] 陈红名. 面向连续动作空间的深度强化学习算法研究[D]. 苏州: 苏州大学, 2020.

[6] 吴俊塔. 基于集成的多深度确定性策略梯度的无人驾驶策略研究[D]. 北京: 中国科学院大学(中国科学院深圳先进技术研究院), 2019.

[7] 柯丰恺, 周唯倜, 赵大兴. 优化深度确定性策略梯度算法[J]. 计算机工程与应用, 2019, 55(7): 151-156, 233.

[8] 赵厚龙. 异步广义优势行动者-评论家及其在自动驾驶中的应用[D]. 青岛: 山东科技大学, 2019.

[9] 张斌, 何明, 陈希亮, 等. 改进 DDPG 算法在自动驾驶中的应用[J]. 计算机工程与应用, 2019, 55(10): 264-270.

[10] 徐博, 周建国, 吴静, 等. 可编程数据平面下基于 DDPG 的路由优化方法[J/OL]. 计算机工程与应用: 1-8[2021-07-24]. http//kns. cnki. net/kcms/detail/11. 2127. TP. 20201221. 1721. 008. html.

[11] 杨薛钰, 陈建平, 傅启明, 等. 基于随机方差减小方法的 DDPG 算法[J/OL]. 计算机工程与应用: 1-10[2021-07-24]. http//kns. cnki. net/kcms/detail/11. 2127. TP. 20201231. 1058. 004. html.

[12] 周友行, 赵晗妘, 刘汉江, 等. 采用 DDPG 的双足机器人自学习步态规划方法[J]. 计算机工程与应用, 2021, 57(6): 254-259.

[13] YANG Y, LI J T, PENG L L. Multi-robot path planning based on a deep reinforcement learning DQN algorithm[J]. CAAI Transactions on Intelligence Technology, 2020, 5(3): 177-183.

[14] 陈亮, 梁宸, 张景异. Actor-Critic 框架下一种基于改进 DDPG 的多智能体强化学习算法[J]. 控制与决策, 2021, 36(1): 75-82.

[15] 唐蕾, 刘广钟. 改进 TD3 算法在四旋翼无人机避障中的应用[J]. 计算机工程与应用, 2021, 57(11): 254-259.