北京航空航天大学学报
Journal of Beijing University of Aeronautics and Astronautics
ISSN 1001-5965,CN 11-2625/V

《北京航空航天大学学报》网络首发论文

题目：基于策略蒸馏的四足机器人步态学习方法
作者：朱晓庆，王涛，阮晓钢，陈江涛，南博睿，毕兰越
DOI：10.13700/j.bh.1001-5965.2023.0069
收稿日期：2023-02-21
网络首发日期：2023-06-02
引用格式：朱晓庆，王涛，阮晓钢，陈江涛，南博睿，毕兰越．基于策略蒸馏的四足机器人步态学习方法[J/OL]．北京航空航天大学学报．https://doi.org/10.13700/j.bh.1001-5965.2023.0069

基于策略蒸馏的四足机器人步态学习方法

朱晓庆[1,2,✉],王涛[1,2],阮晓钢[1,2],陈江涛[1,2],南博睿[1,2],毕兰越[1,2]

(1. 北京工业大学 信息学部,北京 100124; 2. 北京工业大学 计算智能与智能系统北京市重点实验室,北京 100124)

*通信作者 E-mail:alex.zhuxq@bjut.edu.cn

摘要 使得机器人复现高等动物的运动技能是机器学习领域的研究热点。以柔性动作评价(SAC)为代表的强化学习算法在此任务中已取得成功,该框架将策略搜索和状态动作价值函数相结合,在连续控制问题中得到了应用。但智能体使用策略探索是贪婪的,评价网络估算的 Q 值函数却使用低估值。为了使智能体采取更好的策略,本文将策略蒸馏(PD)与 SAC 算法相融合,提出一种策略蒸馏柔性动作评价算法(PDSAC),该算法让智能体使用混合策略进行探索,使强化学习得到的奖励函数收敛速度加快。为验证所提算法有效性,理论证明此算法能提升策略的探索效率并在四足机器人步态学习任务中进行验证。对比仿真实验结果表明,相比 SAC 算法,PDSAC 算法在步态学习任务中可以实现奖励函数值提高 26.7%,同时收敛速度提升 40%。

关键词 强化学习; 策略蒸馏; 混合策略; 好奇心探索策略;步态学习

中图分类号 TP18 文献标识码: A DOI: 10.13700/j.bh.1001-5965.2023.0069

Gait learning method of quadruped robot based on policy distillation

Zhu Xiaoqing[1,2,✉], Wang Tao[1,2], Ruan Xiaogang[1,2], Chen Jiangtao[1,2], Nan Borui[1,2], Bi Lanyue[1,2]

(1.Faulty of Information Technology, Beijing University of Technology, Beijing, 100124, China;

2. Beijing Key Laboratory of Computational Intelligence and Intelligent System, Beijing University of Technology, Beijing, 100124, China)

* E-mail:alex.zhuxq@bjut.edu.cn

Abstract It is a research hotspot in the field of machine learning that the robot can reproduce the motor skill of higher animals. Reinforcement learning algorithms represented by Soft Actor-Critic (SAC) have been successful in the task. The framework combines policy exploration and the state-action value function (Q-value) has been successfully applied in continuous control problems. However, the use of policy exploration by agent is greedy, but the Q-value function estimated by the critic network is usually used for low value. In order to enable agent to adopt better policies, this paper integrates Policy Distillation (PD) and SAC algorithms to propose (Policy Distillation Soft Actor-Critic (PDSAC)) algorithm, which allows agent to explore using hybrid policies and enables the reward function from reinforcement learning to converge faster. To validate the proposed algorithm, Theoretical proof that the PDSAC algorithm improves the efficiency of policy exploration and validation in quadruped robot gait learning task. Comparative simulation results show that the PDSAC can achieve 26.7% improvement in the reward value function and 40% improvement in the convergence speed in the gait learning task compared to the SAC.

Key words Reinforcement learning; Policy distillation; Hybrid policy ; Curiosity exploration policy; Gait learning

如何使四足机器人又快又稳定的运动是近年来的研究热点。强化学习通过智能体与环境交互的状态信息生成策略,并采取相应的行为动作,使期望奖励收敛。其模型被广泛应用于各个领域,如机器人步态控制[1]。四足机器人通过强化学习算法可以在没有任何先验知识的情况下学习步态控制[2]。通过步态信息数据,不断更新优化腿部控制器奖励函数[3]。其中腿部控制器将获取的输入数据映射到低维向量,所以机器人在步态学习中,不需要准确的动力学模型,用较少的操作算法就能学习完整的运动技

收稿日期:2023-02-21

基金项目:国家自然科学基金项目(基金号 62103009);北京市自然科学基金项目(基金号 4202005)

Fund: the National Natural Science Foundation of China (No. 62103009) and the Natural Science Foundation of Beijing (No. 4202005).

网络首发时间: 2023-06-02 17:06:42 网络首发地址: https://kns.cnki.net/kcms2/detail/11.2625.v.20230601.1456.001.html

能[4,5]。

近年来,随着强化学习模型趋于繁琐,策略网络框架也逐渐趋于复杂。将知识蒸馏(Knowledge Distillation, KD)应用于深度强化学习,提出了一种简单而实用的方法称为策略蒸馏(Policy Distillation, PD)。通过辅助智能体策略的生成提升智能体的训练效果,达到深度模型的轻量化[6,7,8]。在 PD 模型中,Rusu 等人提出教师-学生模型,由教师模型学习智能体的状态信息并存储到经验池,通过知识重播将经验池信息向学生模型传递[9]。但教师模型预训练会增加探索成本,并且学生模型获取的信息被教师模型限制。Lai 等人提出将经验池信息向学生模型传递的双策略蒸馏(Dual Policy Distillation, DPD)思想,取消教师模型预训练,使两个学生模型在相同环境中以不同参数进行探索,并通过知识蒸馏提高策略探索效率,弥补教师模型的不足[10]。Zhao 等人通过正则化多个策略分布,实现了多个学生策略互相的知识迁移,增强了策略的泛化能力[11]。Xu 等人提出相似度约束的概念,提升学生策略之间的探索能力及算法稳定性[12,13]。Burda 等人提出随机网络蒸馏模型,使智能体更好的实现局部探索,将内部奖励和外部奖励结合,避免环境中密集奖励不足时,策略蒸馏失效[14]。知识蒸馏和深度强化学习的结合解决了大型强化学习算法中策略网络复杂路径搜索问题,起到了稳定和加速智能体学习策略的效果[15]。

本文基于 DPD 框架,将策略分别表示为原策略和好奇心探索(Curiosity Exploration, CE)策略,其中 CE 探索策略由算法网络中动作值函数和状态值函数共同决定。两个策略之间用相对熵(Kullback-Leibler Divergence, KLD)进行限制[3,16],保证策略在可信的阈值范围,使算法不仅加快智能体的探索速度并且确保奖励值稳步提升直至收敛。

强化学习算法中,动作评价(Actor-Critic)框架的采样效率低,面对复杂控制经常需要上百万次环境交互[17-20],在仿真时增加了不可控性。在无模型强化学习算法中,柔性动作评价算法(Soft Actor Critic, SAC)[21,22]通过添加最大信息熵,让探索策略有更高的随机性。双延迟深度确定性策略梯度[23]算法(Twin Delayed Deep Deterministic policy gradient algorithm, TD3)通过使用较小的状态动作值函数(State-action value function, Q)进行目标网络更新,提高算法的稳定性。但策略探索使用的 Q 值与真实 Q 值偏差过大时,探索效率会变差。

随着深度强化学习所需处理的目标函数维度逐渐增大,在消耗大量的计算存储和时间后才能使智能体得到较高水平的性能。智能体采用何种策略迭代方法将影响最终的奖励收敛速度[24]。其中,普通双策略蒸馏通过两个策略互相提取对智能体探索有利的知识从而优化智能体产生的动作。但两个策略产生的行动差距较大时,较差的策略需从较好的策略学习知识,环境中得到的状态信息使较好的策略不断更新,而较差策略不再更新。此时,双策略蒸馏会退化为单策略蒸馏,造成算力的浪费。针对上述问题,本文提出策略蒸馏柔性动作评价(Policy Distillation Soft Actor-Critic, PDSAC)算法,首先,评价网络通过智能体的动作计算 Q 值,形成原策略和 CE 探索策略,两者组成混合策略;然后,引入 PD 思想,使两者互相蒸馏以更新优化自身策略,从而保持混合策略先进性,加快智能体奖励函数的收敛速度。

本文第 1 节介绍强化学习算法和策略蒸馏的理论和方法,以适应四足机器人步态学习任务。第 2 节证明混合策略的知识迁移就是原策略和好奇心探索策略的知识蒸馏,表明策略的提升可以提高算法的奖励收敛速度。第 3 节给出 PDSAC 算法在四足机器人中的网络框架和代码实现。第 4 节介绍对比实验和消融实验以及结果分析。第 5 节进行结论和展望。

1 强化学习算法和策略蒸馏

1.1 强化学习算法

强化学习算法基于马尔可夫决策过程(Markov Decision Process, MDP)。MDP 由<S,A,P,R>四元组构成。其中 S 表示状态空间,包含智能体内部一系列有限状态 s; A 表示动作空间,智能体与环境交互产生的动作 a 都包含在此空间;P 表示每次状态转移的概率矩阵;R 代表奖励函数。策略 $\boldsymbol{\pi}$ 表示从 S 空间映射到 A 空间的分布函数,包含了智能体在每个状态可能采取的动作。智能体与环境交互时,每一个状态都包含环境的全部信息,通过策略产生动作从而进入下一状态[25]。在某一时刻 t,采取动作 $\boldsymbol{a}_t \sim \boldsymbol{\pi}(\boldsymbol{s}_t)$,使环境进入新状态 $\boldsymbol{s}_{t+1} \sim (\boldsymbol{s}_t, \boldsymbol{a}_t)$,并产生奖励 $r \sim (r_t, \boldsymbol{a}_t)$。

智能体与环境不断进行信息交互，并通过奖励函数调整策略，形成最优策略 $\boldsymbol{\pi}^{*}=\max\left[E_{\boldsymbol{\pi}}\left(\sum_{t=1}^{\tau}\gamma^{t-1}r^{t}\right)\right],\gamma\in[0,1]$，其中 γ 是折扣因子，$\boldsymbol{\tau}\in\{\boldsymbol{s}_1,\boldsymbol{a}_1,r_1\cdots\boldsymbol{s}_t,\boldsymbol{a}_t,r_t\}$ 表示一系列状态动作轨迹。策略 $\boldsymbol{\pi}$ 直接影响智能体的动作输出，从而影响环境中状态信息。可以看出，算法选择合适的策略不仅能加快奖励收敛速度，提高智能体探索效率，还可以使智能体得到更高奖励值。

通过强化学习算法优化策略网络为四足机器人步态学习提供了新的研究方向。策略网络通过当前状态和历史动作预测下一步的环境反馈，实现对不同环境状态的快速适应[26]。

四足机器人的动作控制由一个可训练的神经网络和一个开环控制器组成。开环控制器用于周期信号的生成，四足机器人腿部运动由输出摆动曲线和输出延伸信号组成。通过开环控制器的周期信号估计地形坡度和步态运动的倾斜程度。可训练的策略神经网络通过开环控制器的结果决定下一步动作的输出。如图 1 所示，机器人步态控制由两个信号的矢量和组成。其中开环控制器为策略学习提供了参考轨迹和运动先验，神经网络通过强化学习算法生成的策略决定输出的动作，使得机器人获得更快的动作探索效率和更高的环境奖励值。

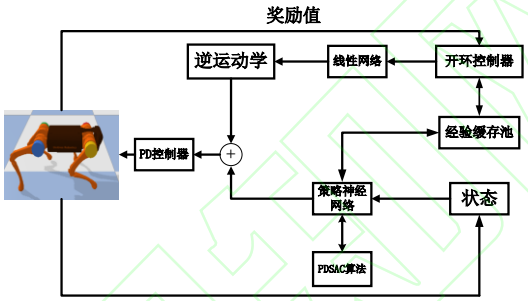


图 1　四足机器人步态控制器
Fig.1 Quadruped robotic gait controller

1.2 柔性动作评价算法

在强化学习算法中，学习目标就是得到一条轨迹 $\boldsymbol{\tau}$，使策略累计奖励值最大。而 SAC 算法中，不仅要求策略累计的奖励值最大，还要求策略的熵值最大[27,28]。智能体在累计奖励中加入正则化项，避免策略落入局部最优解。智能体充分的探索状态空间 S，其中最优策略满足

$$\boldsymbol{\pi}^{*}=\arg\max_{\pi}[\mathrm{E}_{\pi}\left(\sum_{t=1}^{\tau}\gamma^{t-1}r^{t}+\alpha\mathrm{H}\left(\boldsymbol{\pi}\left(\bullet|\boldsymbol{s}_t\right)\right)\right)],\gamma\in[0,1] \tag{1}$$

α 为策略网络熵的系数，表示熵项在算法中的重要程度。式(1)中，熵项

$$\mathrm{H}\left(\boldsymbol{\pi}\left(\bullet|\boldsymbol{s}_t\right)\right)=-\sum_{t}^{\infty}\boldsymbol{p}\left(\boldsymbol{\pi}\left(\bullet|\boldsymbol{s}_t\right)\right)\log\boldsymbol{p}\left(\boldsymbol{\pi}\left(\bullet|\boldsymbol{s}_t\right)\right) \tag{2}$$

表示智能体在每一个状态都计算当前策略 $\boldsymbol{\pi}(\bullet|\boldsymbol{s}_t)$ 的熵值，熵越高，系统的不确定性越强，动作的不确定也越大。$\boldsymbol{p}\left(\boldsymbol{\pi}\left(\bullet|\boldsymbol{s}_t\right)\right)$ 表示状态 s_t 处策略概率分布随机程度，策略越随机，熵值越高。

SAC 算法使用最大熵强化学习，使四足机器人步态优化时比传统的动作评价算法探索能力更强，策略输出的动作更高效，在更短的探索步长使奖励收敛。由式(1)可知，SAC 算法的目标是找到最优策略，使奖励最大。尽管此算法已经取得很好的效果，但仍有以下不足：

(1)当智能体执行策略时，根据高斯采样随机采取探索动作。如图 2 所示，横坐标表示智能体在状态 s 的动作范围，纵坐标为动作值函数(Q 值)，表示在状态 s，策略采取某个动作的价值和采取动作后所有新状态价值与其状态转移概率乘积的和。动作值函数越大，智能体所需的样本需求量越多。图中智能体右边探索时所获 Q 值大，动作范围大，表示右侧的样本需求多，有利于得到更好的策略；在左边探索时样本需求少，不利于策略的更新。为了避免高估，算法会使用 Q 值置信下限 $Q_{\mathrm{LB}}^{i}(\boldsymbol{s},\boldsymbol{a}),\ \ i\in\{1,2\}$ 评估策略，使样本的采样率降低。

(2)虽然 SAC 算法增加熵项鼓励智能体探索,但如果 Q 值得到的策略是次优解,智能体无法选择更好策略,只能执行该策略的采样动作。

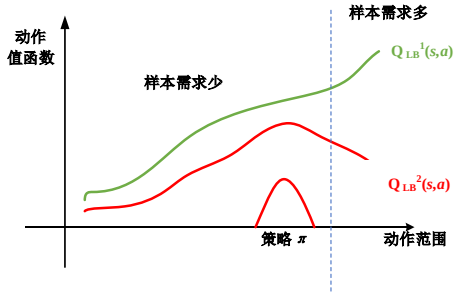


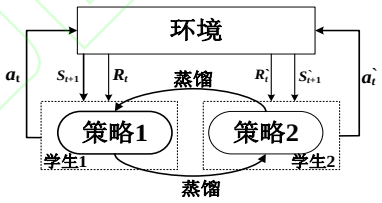
图 2 策略对采样效率的影响
Fig.2 The impact of policy on sampling efficiency

四足机器人通过策略选择动作从而完成步态学习。当奖励函数收敛时,步长和奖励数值可以反映算法使用策略的优劣程度。通过不断的策略更新,四足机器人生成柔顺协调的运动步态。奖励值收敛速度间接表示步态优化程度。当算法选择了最优策略,奖励函数收敛,机器人步态稳定。由于策略的效率直接影响奖励收敛和行走步长。选择合适的强化学习算法可以使策略神经网络得到更优秀的策略,从而加速步态规划。

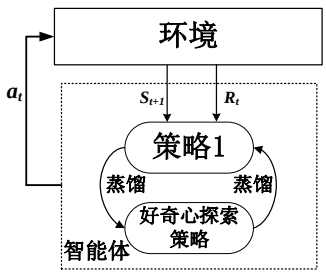
1.3 策略蒸馏

知识蒸馏由 Hinton 提出,属于迁移学习的分支,利用大模型训练小模型将大模型的信息传递到小模型中,通过学生模型模仿教师模型输出进行知识迁移的学习方法[29,30]。

策略蒸馏由 Czarnecki 等人提出,将知识蒸馏应用于强化学习过程[6]。该方法将预训练的评价网络模型作为教师模型,并将智能体学习到的状态,动作分布概率,奖励作为知识存储到经验池(Reply Memory)中,在训练学生模型时经验池中的“知识”将作为指导学生的依据。师生策略的知识迁移可以用于提升网络性能,加快奖励收敛,训练复杂任务网络。Rusu 等人使用了监督回归训练学生模型,使其与教师模型有相同的分布;Lai 等人构建了双策略蒸馏框架,通过在相同环境中探索不同的策略并通过两个学生策略互相进行知识迁移,从而摆脱了教师模型的限制。如图 3(a)所示。



(a) 双策略蒸馏框架



(b) 基于好奇心探索策略的混合策略蒸馏框架

图 3 策略蒸馏框架
Fig.3 Policy distillation framework

本文在双策略蒸馏框架的基础上,引入好奇心探索策略,使策略在提升过程中更快的搜集环境反馈

信息，加速智能体训练。如图 3(b)所示。

相互蒸馏训练深度神经网络提高性能由 Ying 等人提出[31]。其思想是让一组未经训练的学生模型同时进行知识的学习,通过相互蒸馏补充多样化信息使原始组合变为性能更好的新组合。

本文采用的蒸馏方法为相互蒸馏中的深度互学习(Deep Mutual learning, DML)。如图 4(a)所示,DML 使用两个网络,智能体与环境交互产生的环境信息表示为 $\boldsymbol{X}=\{\boldsymbol{X}_i\}_{i=1}^{N}$,其中 N 代表经验池中的状态数量。X_i 表示每个状态 s 包含的全部策略动作信息。X_i 进入神经网络Θ产生的策略概率计算公式为

$$\boldsymbol{P}_1(\boldsymbol{X}_i)=\frac{\exp(z_1)}{\sum\exp(z_1)} \tag{3}$$

其中 z_1 是神经网络 softmax 层的输出,通过使用 KL 散度使两个网络的策略能互相蒸馏，组合策略融合了策略信息,智能体可以根据组合策略选择动作。P_1 和 P_2 为数据的概率分布。P_1 到 P_2 的 KL 散度为：

$$\boldsymbol{D}_{\mathrm{KL}}(\boldsymbol{p_2}\parallel\boldsymbol{p_1})=\sum_{i=1}^{N}\boldsymbol{p_2}(\boldsymbol{X}_i)\log\frac{\boldsymbol{p_2}(\boldsymbol{X_i})}{\boldsymbol{p_1}(\boldsymbol{X}_i)} \tag{4}$$

在 $\boldsymbol{D}_{\mathrm{KL}}(\boldsymbol{p_2}\parallel\boldsymbol{p_1})$中, P_1 把 P_2 视为数据的真实概率分布,使用 KL 散度使 P_1 的分布近似 P_2 的分布。P_2 到 P_1 的 KL 散度同理。DML 以相互学习的方式训练各个状态信息，通过互相补充多样化的策略信息生成更强的算法模型。

在 MDP 特征序列任务中,上一步状态的策略知识传递给后续状态进行学习,减少了新旧网络在经验池中信息差异。本文提出的 PDSAC 算法,如图 4(b)所示, 环境无需同时给两个学生模型状态信息,只需通过一个学生模型产生的策略状态信息,CE 策略通过组合策略相对距离 Δ 产生,两者构成混合策略。在四足机器人的策略优化中,策略的输出是四条腿期望的关节向量。通过引入 DML,好奇心探索策略和原策略交替优化策略神经网络最终找到每个状态的最优策略。原策略与智能体期望策略的损失函数表示为

$$L_{C_1}=-\sum_{i=1}^{N}I(\boldsymbol{\pi},\boldsymbol{\pi}^*)\rho\left(\boldsymbol{\pi}(\bullet\mid\boldsymbol{s}_t)\right)\log(p_i(\boldsymbol{\pi}(\bullet\mid\boldsymbol{s}_t))) \tag{5}$$

其中 $I(\boldsymbol{\pi},\boldsymbol{\pi}^*)=\begin{cases}0 & \boldsymbol{\pi}=\boldsymbol{\pi}^*\\ 1 & \boldsymbol{\pi}\neq\boldsymbol{\pi}^*\end{cases}$,表明如果此刻状态 s 为最优策略,误差为 0，否则将一直探索。

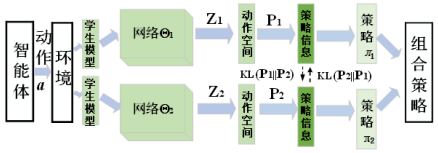
$\rho\left(\boldsymbol{\pi}\left(\bullet\mid\boldsymbol{s}_t\right)\right)$为 s_t 处策略概率分布随机程度, $p_i(\boldsymbol{\pi}(\bullet\mid\boldsymbol{s}_t))$ 为 s_t 处某一策略的概率。同理,好奇心策略的交叉熵误差为 $\boldsymbol{L}_{C_2}$。网络Θ1 和Θ2 的损失函数定义为：

$$\begin{cases}L_{\Theta_1}=L_{C_1}+D_{KL}(\boldsymbol{p_2}\parallel\boldsymbol{p_1})\\ L_{\Theta 2}=L_{C_2}+D_{KL}(\boldsymbol{p_1}\parallel\boldsymbol{p_2})\end{cases} \tag{6}$$

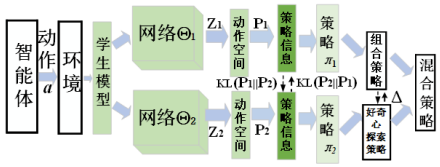
网络参数梯度更新公式为：

$$\begin{cases}\Theta_1\leftarrow\Theta_1+\gamma_{1,t}\dfrac{\partial L_{\Theta_1}}{\partial_{\Theta_1}}\\ \Theta_2\leftarrow\Theta_2+\gamma_{2,t}\dfrac{\partial L_{\Theta_2}}{\partial_{\Theta_2}}\end{cases} \tag{7}$$

γ_t 包含了当前状态的所有信息,每次有新环境信息进入经验池中后,根据式(3)更新策略概率,式(7)更新网络参数。由于智能体采取优秀动作是提升策略的关键，引入好奇心探索策略后，扩展了策略的输入和熵值较大的动作输出。经验缓冲池可以提高策略的采样效率。使机器人步态运动技能的学习加快,奖励收敛速度提前。



(a) 普通双策略深度互学习



(b) 引入好奇心探索策略的深度互学习

图 4 深度学习相互蒸馏示意图
Fig.4 Deep learning mutual distillation schematic

2 PDSAC 算法理论分析

PDSAC 算法使用 Q 值评估的原策略和好奇心探索策略构成混合策略,通过策略蒸馏的方式推进策略的更新。2.1 节介绍好奇心探索策略的概率分布及应用。2.2 节介绍了混合策略可以通过优势函数实现稳步提升,使智能体策略不断优化。2.3 节证明混合策略之间的知识迁移就是原策略和好奇心探索策略之间的知识蒸馏。2.4 节介绍混合策略在四足机器人中如何进行策略更新。

2.1 好奇心探索策略

原策略 $\boldsymbol{\pi}$ 和好奇心探索策略 $\boldsymbol{\pi}_{\text{ce}}$ 之间的相对距离定义为 $\Delta = D(\boldsymbol{\pi}(\bullet|\boldsymbol{s}), \boldsymbol{\pi}_{\text{ce}}(\bullet|\boldsymbol{s}))$,表示 $\boldsymbol{\pi}$ 在状态 s 处动作分布范围的偏移。在四足机器人步态策略的形成中,使用 KL 散度 $\boldsymbol{D}_{KL}(\boldsymbol{p}_1 \| \boldsymbol{p}_2), \boldsymbol{D}_{KL}(\boldsymbol{p}_2 \| \boldsymbol{p}_1) \le \Delta$ 约束两个策略之间的相对距离,避免策略偏移过大使 $\boldsymbol{\pi}_{ce}$ 和 $\boldsymbol{\pi}$ 之间的相互蒸馏收集无用的动作经验，对策略的学习没有实质性帮助。

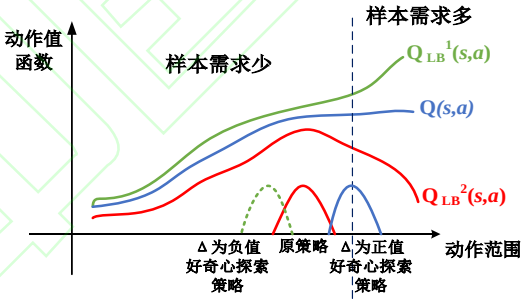


图 5 PDSAC 算法的应用
Fig.5 Application of the PDSAC algorithm

如图 5 所示,采用原策略 $\boldsymbol{\pi}$,收集的样本信息较少。但采用 $\boldsymbol{\pi}_{\text{ce}}$ 采样,当相对距离 Δ 是正值时,智能体收集的样本使下一个状态的动作价值函数变大,采样效率加快;如果 Δ 是负值,继续使用原策略,不会降低采样效率。两者通过式(4)完成策略信息共享。

$\boldsymbol{\pi}_{\text{ce}}$ 服从正态分布 $\boldsymbol{N}(\boldsymbol{\mu}_{\text{ce}}, \boldsymbol{\sigma}_{\text{ce}})$,其中,均值定义为

$$\boldsymbol{\mu}_{\text{ce}} = \underset{\substack{\boldsymbol{p}(\boldsymbol{\pi}(\bullet|\boldsymbol{s}_t)):\\ \boldsymbol{D}_{KL}(\boldsymbol{p}_1\|\boldsymbol{p}_2), \boldsymbol{D}_{KL}(\boldsymbol{p}_2\|\boldsymbol{p}_1) \le \Delta}}{\operatorname{argmax}} E_{\boldsymbol{a} \sim \boldsymbol{N}(\boldsymbol{\mu},\ \delta)} \left[\frac{1}{2} \left(\boldsymbol{Q}_{\text{LB}}^{1}(\boldsymbol{s},\ \boldsymbol{a}) + \boldsymbol{Q}_{\text{LB}}^{2}(\boldsymbol{s},\ \boldsymbol{a}) \right) \right] \tag{8}$$

式中 $\boldsymbol{Q}_{\text{LB}}^{1}$ 和 $\boldsymbol{Q}_{\text{LB}}^{2}$ 是评价神经网络计算的两个置信下限。$\boldsymbol{Q}_{\text{LB}}^{\{1,2\}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ 表示智能体在两个评价网络置信下限中使用较小值,满足:

$$Q_{\mathrm{LB}}^{\{1,2\}}(s_t,a_t)\leftarrow \mathrm{R}(s_t,a_t)+\gamma min\left(Q_{LB}^{1}(s_{t+1},a),Q_{LB}^{2}(s_{t+1},a)\right) \tag{9}$$

式中 $a\sim\pi(\bullet|s_t)$ 表示策略采取的动作,π_{ce} 与 π 的方差 σ 相同。

2.2 策略混合

将 π_{ce} 和原策略混合,如图 4(b)所示，两者之间用相对距离进行限制。在 PDSAC 算法中,两个策略通过知识蒸馏完成信息交互。混合策略表示为：

$$\tilde{\pi}(\bullet|s)=\begin{cases}\pi(\bullet|s) & \xi^{\pi}(s)\le 0\\ \pi_{\mathrm{ce}}(\bullet|s) & \xi^{\pi}(s)>0\end{cases} \tag{10}$$

式中,$\xi^{\pi}(s)$ 是混合策略选择 π_{ce} 或 π 的条件,表示在状态 s 处,π_{ce} 相对于 π 动作值优势。表达式为 $\xi^{\pi}(s)=V^{\pi_{ce}}(s)-V^{\pi}(s)$。在 MDP 中,$V(s)$表示在状态 s 处所有动作值函数在策略下的期望[32,33]。$V^{\pi}(s)$ 和 $V^{\pi_{ce}}(s)$ 表示 π 和 π_{ce} 在状态 s 处的动作值函数。由于相对位移 Δ 方向是随机的,所以混合策略选取策略的标准是在状态 s 处,选择 $V(s)$较大的策略。定理 1 证明混合策略的价值函数是稳步提升的。

定理 1. 对于任意状态 $s\in S$,式(10)定义的混合策略会稳步提升,即当 $n\ge 0$ 时,满足 $V_{n+1}\ge V_n$。n 表示状态价值函数在当前状态处的迭代次数。

证明：

由于优势函数 $\xi^{\pi}(s)$ 表示为 π_{ce} 和 π 在环境状态 s 处智能体采取动作的价值期望差值。混合策略 $\tilde{\pi}(s_i)$ 的动作值函数 $V^{\tilde{\pi}}$ 满足

$$\begin{cases}V^{\tilde{\pi}}=V^{\pi} & \xi^{\pi}(s)\le 0\\ V^{\tilde{\pi}}=V^{\pi_{ce}} & \xi^{\pi}(s)>0\end{cases} \tag{11}$$

由于 $V^{\tilde{\pi}}$ 基于 π 和 π_{ce} 中状态价值较大的策略。通过式(11),在状态 s 处,混合策略的价值函数最大,$\tilde{\pi}(s_i)$ 为此状态的最优策略。

当四足机器人处于特定状态 s,定义此状态 $s_t\in S$,则下一状态为 s_{t+1}。此时,该状态的价值函数定义为：

$$V_n(s_i)=\begin{cases}V^{\tilde{\pi}(s_i)}(s_i) & n=0\\ \mathrm{E}_{(s_{i+1},a)\sim\tilde{\pi}}\left[r_i+\gamma V_{n-1}(s_{i+1})\right] & n\ge 1\end{cases} \tag{12}$$

当 n=0 时,$V_n(s_i)=V^{\tilde{\pi}(s_i)}(s_i)$。首先证明,对于任意状态 $s_t\in S$,$V_1(s_i)\ge V_0(s_i)$。

$$\begin{aligned}V_1(s_i)&=\mathrm{E}_{(s_{i+1},a_{i+1})\sim\tilde{\pi}(s_i)}\left[r_i+\gamma V_0(s_{i+1})\right]\\ &=\sum_{s_{i+1},r_i}p^{\tilde{\pi}}(s_{i+1},r_i\mid s_t)\left[r_i+\gamma V_0(s_{i+1})\right]\\ &=\sum_{s_{i+1},r_i}p^{\tilde{\pi}}(s_{i+1},r_i\mid s_t)\left[r_i+\gamma V_0^{\tilde{\pi}(s_{i+1})}(s_{i+1})\right]\\ &\ge\sum_{s_{i+1},r_i}p^{\tilde{\pi}}(s_{i+1},r_i\mid s_t)\left[r_i+\gamma V_0^{\tilde{\pi}(s_i)}(s_{i+1})\right]\\ &=\sum_{s_{i+1},r_i}p^{\tilde{\pi}(s_i)}(s_{i+1},r_i\mid s_t)\left[r_i+\gamma V_0^{\tilde{\pi}(s_i)}(s_{i+1})\right]\\ &=V^{\tilde{\pi}(s_i)}(s_i)\\ &=V_0(s_i)\end{aligned} \tag{13}$$

式中 p 表示状态转移概率矩阵。

当 $n\ge 1$ 时,对于任意的状态 $s_t\in S$,证明 $V_n(s_i)\ge V_{n-1}(s_i)$

$$\begin{aligned} V_{n+1}(\boldsymbol{s}_i) &= \mathrm{E}_{(\boldsymbol{s}_{i+1},\boldsymbol{r}_i)\sim\tilde{\boldsymbol{\pi}}(\boldsymbol{s}_i)}[\boldsymbol{r}_i+\gamma V_n(\boldsymbol{s}_{i+1})] \\ &= \sum_{\boldsymbol{s}_{i+1},\boldsymbol{r}_i} \boldsymbol{p}^{\tilde{\boldsymbol{\pi}}}(\boldsymbol{s}_{i+1},\boldsymbol{r}_i\mid\boldsymbol{s}_i)[\boldsymbol{r}_i+\gamma V_n^{\tilde{\boldsymbol{\pi}}(\boldsymbol{s}_{i+1})}(\boldsymbol{s}_{i+1})] \\ &\geq \sum_{\boldsymbol{s}_{i+1},\boldsymbol{r}_i} \boldsymbol{p}^{\tilde{\boldsymbol{\pi}}}(\boldsymbol{s}_{i+1},\boldsymbol{r}_i\mid\boldsymbol{s}_i)[\boldsymbol{r}_i+\gamma V_{n-1}^{\tilde{\boldsymbol{\pi}}(\boldsymbol{s}_{i+1})}(\boldsymbol{s}_{i+1})] \\ &= \sum_{\boldsymbol{s}_{i+1},\boldsymbol{r}_i} \boldsymbol{p}^{\tilde{\boldsymbol{\pi}}(\boldsymbol{s}_i)}(\boldsymbol{s}_{i+1},\boldsymbol{r}_i\mid\boldsymbol{s}_i)[\boldsymbol{r}_i+\gamma V_{n-1}(\boldsymbol{s}_{i+1})] \\ &= V_n(\boldsymbol{s}_i) \end{aligned} \tag{14}$$

通过归纳法,可以得出在连续的状态空间,对于任意的 n 和对于所有的状态 s,满足 $V_{n+1}(\boldsymbol{s})\geq V_n(\boldsymbol{s})$。

证毕。

混合策略的优势为策略迭代过程中,智能体不再以单一的策略完成迭代,而是从 $\boldsymbol{\pi}$ 和 $\boldsymbol{\pi}_{\mathrm{ce}}$ 选取较为优秀的策略作为智能体下一个状态采取的策略。上式从理论证明 PDSAC 算法在更新后的混合策略优于当前策略,即策略是不断提升的。通过比较两个策略的动作值函数决定智能体采用何种策略。如果使用 $\boldsymbol{\pi}_{\mathrm{ce}}$,将 $\boldsymbol{\pi}$ 蒸馏为 $\boldsymbol{\pi}_{\mathrm{ce}}$ 进入下一状态。如果使用 $\boldsymbol{\pi}$,则将 $\boldsymbol{\pi}_{\mathrm{ce}}$ 舍弃进入下一状态。

2.3 优势策略蒸馏

算法的数据训练是从经验池中选择环境状态信息批量的输入给神经网络,并计算输出损失函数,通过反向网络传播进行梯度更新。而 PDSAC 算法通过引入 $\boldsymbol{\pi}_{\mathrm{ce}}$ 和 $\boldsymbol{\pi}$ 的知识迁移,使神经网络加速获得策略最优解。

定理 2 证明混合策略 $\tilde{\boldsymbol{\pi}}$ 将知识迁移到原策略 $\boldsymbol{\pi}$,等价于 $\boldsymbol{\pi}_{\mathrm{ce}}$ 向 $\boldsymbol{\pi}$ 的策略蒸馏。

定理 2. 从混合策略 $\tilde{\boldsymbol{\pi}}$ 中进行知识迁移到原策略 $\boldsymbol{\pi}$,等价于最小化如下目标函数:

$$J_{\boldsymbol{\pi}} = E_{\boldsymbol{s}:\boldsymbol{\pi}_{ce}}[D\left(\boldsymbol{\pi}\left(\bullet\mid\boldsymbol{s}\right),\boldsymbol{\pi}_{ce}\left(\bullet\mid\boldsymbol{s}\right)\right)\tau(\xi^{\boldsymbol{\pi}_{ce}}(\boldsymbol{s})>0)] \tag{15}$$

其中 $D(\boldsymbol{\pi}(\bullet\mid\boldsymbol{s}),\boldsymbol{\pi}_{ce}(\bullet\mid\boldsymbol{s}))$ 表示原策略和好奇心探索策略之间的相对距离 Δ。$\tau\left(\xi^{\boldsymbol{\pi}}\right)$ 是一个函数,当函数内的优势函数 $\xi^{\boldsymbol{\pi}}(\boldsymbol{s})=V^{\boldsymbol{\pi}_{ce}}(\boldsymbol{s})-V^{\boldsymbol{\pi}}(\boldsymbol{s})$ 大于 0 设定为 1;优势函数小于 0 设定为 0。优势函数 $\xi^{\boldsymbol{\pi}}(\boldsymbol{s})$ 大于 0 也说明在当前状态,$\boldsymbol{\pi}_{\mathrm{ce}}$ 所能收获的期望奖励要大于 $\boldsymbol{\pi}$,好奇心探索策略对下一时刻的状态有利。

证明:

由于 $\boldsymbol{\pi}_{\mathrm{ce}}$ 和 $\boldsymbol{\pi}$ 都是由 Q 值得,所以在策略蒸馏中忽略环境的影响。从混合策略 $\tilde{\boldsymbol{\pi}}$ 开始的策略蒸馏过程表示为:

$$\begin{aligned} J_{\pi} &= \mathrm{E}_{\boldsymbol{s}\sim\tilde{\boldsymbol{\pi}}}[D\left(\boldsymbol{\pi}\left(\bullet\mid\boldsymbol{s}\right),\tilde{\boldsymbol{\pi}}\left(\bullet\mid\boldsymbol{s}\right)\right)] \\ &= \sum_{\boldsymbol{s}\sim\boldsymbol{p}_{\tilde{\pi}}(\bullet)} D\left(\boldsymbol{\pi}\left(\bullet\mid\boldsymbol{s}\right),\tilde{\boldsymbol{\pi}}\left(\bullet\mid\boldsymbol{s}\right)\right) \\ &= \sum_{\boldsymbol{s}\sim\boldsymbol{p}_{\tilde{\pi}}(\bullet);\xi^{\pi}(\boldsymbol{s})>0} D\left(\boldsymbol{\pi}\left(\bullet\mid\boldsymbol{s}\right),\boldsymbol{\pi}_{\mathrm{ce}}\left(\bullet\mid\boldsymbol{s}\right)\right) \\ &\quad + \sum_{\boldsymbol{s}\sim\boldsymbol{p}_{\tilde{\pi}}(\bullet);\xi^{\pi}(\boldsymbol{s})\leq 0} D\left(\boldsymbol{\pi}_{\mathrm{ce}}\left(\bullet\mid\boldsymbol{s}\right),\pi\left(\bullet\mid\boldsymbol{s}\right)\right) \\ &= \mathrm{E}_{\boldsymbol{s}\sim\boldsymbol{\pi}_{\mathrm{ce}}}[\mathrm{D}\left(\boldsymbol{\pi}\left(\bullet\mid\boldsymbol{s}\right),\boldsymbol{\pi}_{\mathrm{ce}}\left(\bullet\mid\boldsymbol{s}\right)\right)\tau\left(\xi^{\boldsymbol{\pi}_{\mathrm{ce}}}(\boldsymbol{s})>0\right)] \end{aligned} \tag{16}$$

证毕。

在策略蒸馏后,智能体根据混合策略采样动作。根据上述证明,当 $\boldsymbol{\pi}_{\mathrm{ce}}$ 有优势时,$\boldsymbol{\pi}$ 被蒸馏,使两个策略保持相同概率分布。当 $\boldsymbol{\pi}_{\mathrm{ce}}$ 没有优势时,函数将自动屏蔽策略蒸馏过程,加快策略更新速率。

2.4 混合策略更新

PDSAC 算法通过不断的策略蒸馏,使到达步态稳定及奖励收敛的步长变短,提高智能体探索效率。$\boldsymbol{\pi}_{\mathrm{ce}}$ 使智能体选择行动时,不再局限于原策略,而通过优势函数选择更好的策略。$\boldsymbol{\pi}_{\mathrm{ce}}$ 通过深度互学习将

策略概率信息传递给 $\boldsymbol{\pi}$,使策略在不断提升的过程中探索出最优策略。其中,下一个状态的动作价值函数由当前状态使用混合策略进行高斯采样的动作形成。状态价值的迭代方式为:

$$V_{i+1}(\boldsymbol{s}_t)\leftarrow \max_{\boldsymbol{a}\sim\boldsymbol{\pi}(\cdot|\boldsymbol{s})}\sum_{\boldsymbol{s}_{t+1}}\boldsymbol{p}_{\boldsymbol{\tau}}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t,\boldsymbol{a})[R(\boldsymbol{s}_t,\boldsymbol{a},\boldsymbol{s}_{t+1})+V_i(\boldsymbol{s}_{t+1})] \tag{17}$$

其中 $V_i(\bullet)$ 和 $V_{i+1}(\bullet)$ 表示当前状态和下一个状态的价值函数。 $\boldsymbol{p}_{\boldsymbol{\tau}}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t,\boldsymbol{a})$ 表示策略的状态转移概率矩阵。在状态 s 的最优策略价值函数 $V^{\boldsymbol{\pi}^*}$ 和混合策略价值函数 $V^{\tilde{\boldsymbol{\pi}}}$,满足 $V^{\boldsymbol{\pi}^*}\geq V^{\tilde{\boldsymbol{\pi}}}$。而混合策略的选择基于 $\boldsymbol{\pi}_{\text{ce}}$ 和 $\boldsymbol{\pi}$ 的优势函数,在任何时刻,都会采取使 $V^{\tilde{\boldsymbol{\pi}}(s)}$ 最大的值。所以在每个状态,混合策略都会选择更优的策略传给智能体的动作网络,并通过策略蒸馏使策略在提升中不断逼近最优策略,最终使奖励函数最大且收敛。

3 PDSAC 算法网络框架

3.1 网络结构

深度学习中神经网络主要功能是进行卷积,池化,激活。神经元个数过多,会造成算法的过拟合,导致探索能力下降;神经元个数过少,浅层的学习能力下降,探索的效率降低[34,35]。如图 6 所示,PDSAC 算法中,中间神经元个数设置为 256 个。动作网络包含动作平均支路和动作标准支路,并使用了 $f(x)=\log(1+e^x)$ 的激活函数使标准支路的输出恒为正值。在 PDSAC 算法中混合策略做为输出,使四足机器人根据混合策略进行动作高斯采样并收集经验元组<s_i,a_i,r_i,s_{i+1}>,然后继续进行策略的更新,重复采样动作,直到输出的策略是最优解。

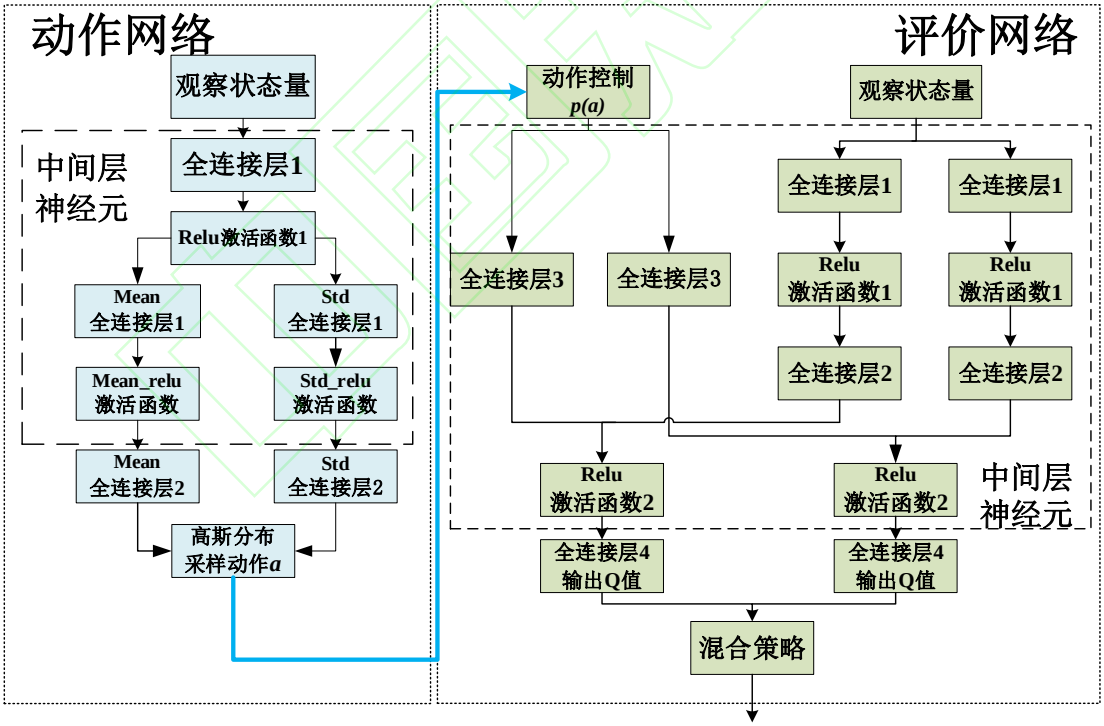


图 6 PDSAC 算法策略神经网络
Fig.6 PDSAC algorithm policy neural network

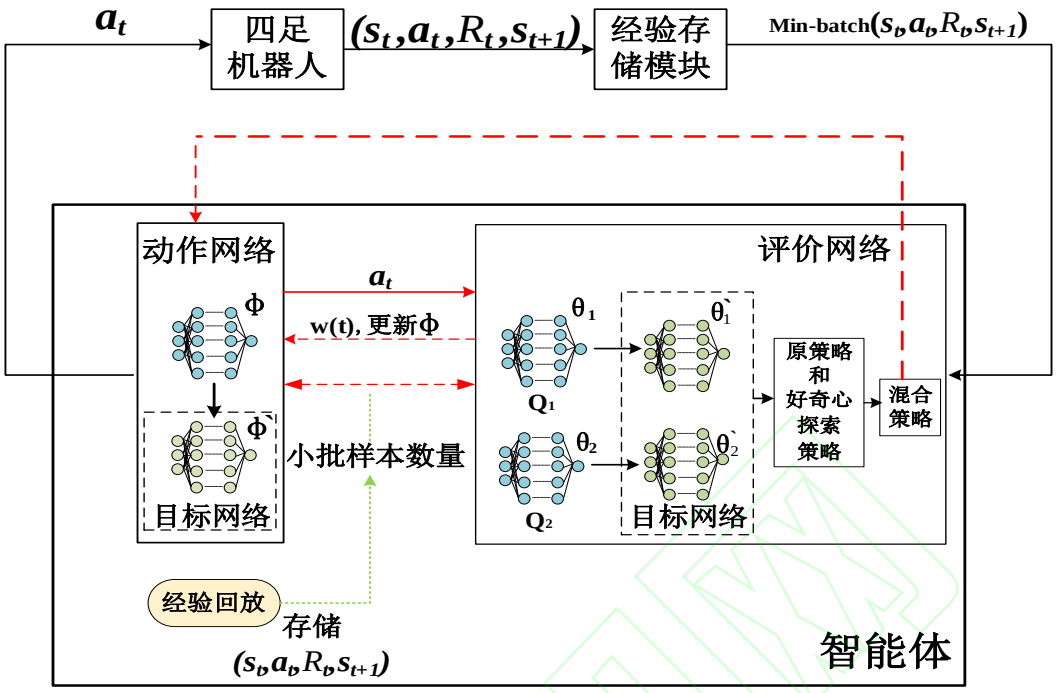


图 7 PDSAC 算法与环境交互
Fig.7 PDSAC algorithms interact with the environment

3.2 混合策略与四足机器人交互

混合策略通过式(17)完成更新和经验池数据收集并进入下一个状态,直到价值函数 $V(s)$收敛。在算法训练过程中,前期需要大量的探索使智能体获得更多的策略进行学习,尽可能的保证不遗漏每一个状态;后期进行相对少的探索使策略逐渐趋于稳定。

如图 7 所示,动作网络进行策略更新和动作的输出。评价网络通过动作价值网络及对应的目标网络计算 Q 值,使用混合策略提高采样效率。引入最大熵和动作重参数化,提高智能体的空间探索能力[13]。其中策略参数更新用 ϕ 表示,评价网络参数更新用 θ 表示。

3.3 PDSAC 算法代码实现

表1 PDSAC算法代码
Table 1 PDSAC Algorithm Code

输入：动作网络参数 ϕ，ϕ'，评价网络参数 θ_1，θ_2，初始状态信息 s_t。

输出：目标网络参数 θ_1' 和 θ_2'。

FOR 每一个时间步长。

Step1. 在状态 s_t 处, 从混合策略 $\tilde{\pi}$ 选择 $a \sim \tilde{\pi}(a_t|s_t)$, 智能体执行动作 a_t, 并获得奖励 R_t, 进入下一个状态 s_{t+1}。

Step2.在经验池中存储样本 (s_t,a_t,R_t,s_{t+1}), 经验池满足 $B_k \leftarrow B_k \cup \{s_t,a_t,R_t,s_{t+1}\}$。

Step3.从经验池随机抽取 40 个样本数量 40*(s_t,a_t,R_t,s_{t+1}), 用式(4)进行策略蒸馏, 式(17)进行混合策略更新。

FOR 对于每一次策略更新,都将更新以下参数

Step1. 更新 Q 网络参数 $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}\theta_i J_Q(\theta_i), i \in \{1,2\}$。

Step2. $\boldsymbol{\pi}$ 与 $\boldsymbol{\pi}_{\text{ce}}$ 组成混合策略 $\tilde{\pi}$ 。

Step3.用混合策略更新动作网络参数 $\phi \leftarrow \phi - \lambda_{\tilde{\pi}} \hat{\nabla}_{\phi} J_{\tilde{\pi}}(\phi)$

Step4.更新信息熵系数 $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_{\alpha} J(\alpha)$

Step5.更新评价网络的参数 $\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i', i \in \{1,2\}$ 。

END

END

首先动作网络和评价网络进行初始化,使智能体得到初始环境信息。第一个 FOR 循环中,智能体首次进行环境交互,由于没有 Q 值函数的更新,$\boldsymbol{\pi}_{\text{ce}}$ 和 $\boldsymbol{\pi}$ 是相同的。智能体采取动作进入下一个状态后,评价网络根据新的状态信息计算 Q 值,此时 $\boldsymbol{\pi}_{\text{ce}}$ 和 $\boldsymbol{\pi}$ 有相对位移 Δ,两者由式(10)构成混合策略。并通过式(17)使混合策略拥有最优价值信息。智能体采取行动后,将得到的状态信息和反馈奖励都存储在经验池中,并在达到设定步长后进行更新迭代。第二个 FOR 循环是进行网络参数的迭代,与 SAC 算法的区别是 PDSAC 算法是使用混合策略进行动作网络的更新,这将促进智能体的探索,从而得到更好的样本值,使收敛速度加快。

4 实验验证

为了验证本文提出的 PDSAC 算法有效性,在机器人物理模拟环境 Pybullet 中进行四足机器人步态学习任务的训练,并绘制图表观察奖励函数的收敛速率。

4.1 实验环境

Pybullet 可用于机器人的仿真模拟。为了有效地评估 PDSAC 算法性能,四足机器人在 49 维的向量空间中包含全部状态信息。如图 8 所示,28 维表示 4 条腿的维度信息,每条腿有 7 个自由度。其中髋关节由水平旋转和俯仰两个自由度,膝关节使腿有灵活度和整体稳定控制,腕部实现俯仰的一个自由度,大腿和小腿包含电机控制信息。这些信息通过策略产生的动作互相配合,使四足机器人实现行走步态。向量空间中也包含惯性测量单元(Inertial Measurement Unit, IMU)获得的翻滚,俯仰,偏航的 3 维向量空间和关节角速度向量空间。

算法策略的优劣通过四足机器人步态学习程度体现。算法的奖励函数为:

$$r_t = (\boldsymbol{P}_{t-1} - \boldsymbol{P}_t) * \boldsymbol{D}_t * (1-\lambda) \tag{18}$$

式中,D_t 表示期望的方向,P_t 表示当前机器人的位置,P_{t-1} 表示上一时刻的位置,权重 $\lambda \sim (0,1)$ 决定了能量消耗在奖励函数中的重要性。通过激励四足机器人沿着策略期望方向运动,使奖励值增加。奖励函数也表明,训练步长的大小可以反应策略的优劣程度。训练步长越小,表明算法选择的策略越好。

策略输出的动作<a_1,a_2...a_{12}>是维度为 12 维向量空间。每一维对应腿部关节所需角度信息。算法训练的目标就是找到最优策略,根据当前环境状态信息,输出动作后,智能体保持稳定的运动步态。

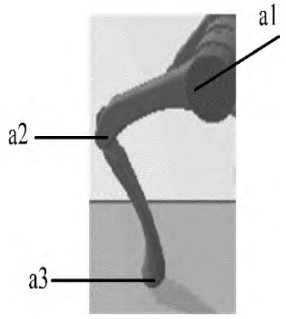


图 8 四足机器人单腿示意图

Fig.8 Schematic diagram of quadruped robot with one leg

4.2 对比实验

SAC 算法和 PDSAC 算法的所有超参数保持一致,并且每种算法与环境交互 100 万次时间步。由于 $\boldsymbol{\pi}$ 与 $\boldsymbol{\pi}_{ce}$ 的相对位移 Δ 会影响奖励值的收敛程度。Δ 较小时,两者距离过近,Q 值会陷入局部次优解,策略迭代需要的价值函数偏低甚至不如原策略;Δ 偏大时,混合策略与评价网络计算的 Q 值相关性降低,使机器人运动不平稳,探索的时间加长。根据经验,将相对位移设置为 0.5。

四组机器人使用高斯分布进行动作采样,输入是环境的状态,输出是多维高斯分布动作向量。图 9 为不同算法的奖励值对比,其中横轴为训练步长,纵轴为奖励值。PDSAC 算法在 30 万步时就可以使奖励收敛,并且获得高奖励值(3800);SAC 算法需要 50 万步才会进入收敛区,奖励值较低(3000)且变化幅度大。而 PDSAC 算法会一直保持高奖励值。数据显示,PDSAC 算法在收敛速度上比 SAC 算法提升了 40%,奖励函数提高 26.7%。

PDSAC 算法的 $\boldsymbol{\pi}_{ce}$ 与环境的交互对数据有积极影响,尤其在经验池的数据较少时,在训练前期表现为奖励的快速提升;在 15 万步之后,随着经验池的状态信息增多,策略的分布逐渐趋于稳定,探索性下降,但当一个策略受到劣势数据影响时,可以快速的进行知识蒸馏。

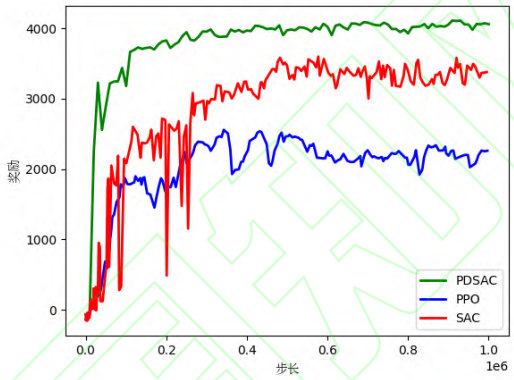


图 9 不同算法的奖励值对比

Fig.9 Comparison of reward values for different algorithm

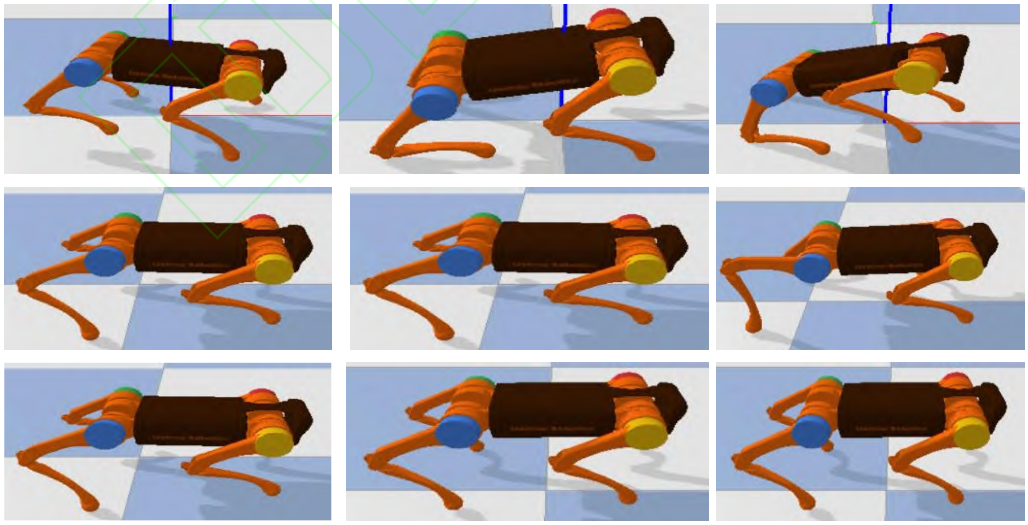


图 10 机器人在前期中期后期的运动步态,其中第一行为前期,第二行为中期,第三行为后期

Fig.10 Gait of the robot in the early, middle and late stages of its movement, the first of the pictures is the first period, the second of the pictures is the middle movement, the third of the pictures is the late stages

图 10 为四足机器人在 PDSAC 算法下的步态学习过程截图。如图 10 第一行三个子图所示,机器人

在训练初期的 5 万步,四条腿摆动幅度大,质心高度一直在无规则变化；如图 10 第二行三个子图所示,在训练中期的 20 万步时,机器人腿部不再大范围摆动,四条腿实现规律向前运动,质心趋于稳定；如图 10 第三行三个子图所示,在训练后期的 30 万步时,机器人四条腿呈现节律运动,机器人整体保持匀速前进且保持姿态平稳,成功完成步态学习任务。验证了策略蒸馏的引入,可以让四足机器人比原有算法更快的完成策略的探索,加强了算法在训练后期的稳定性。

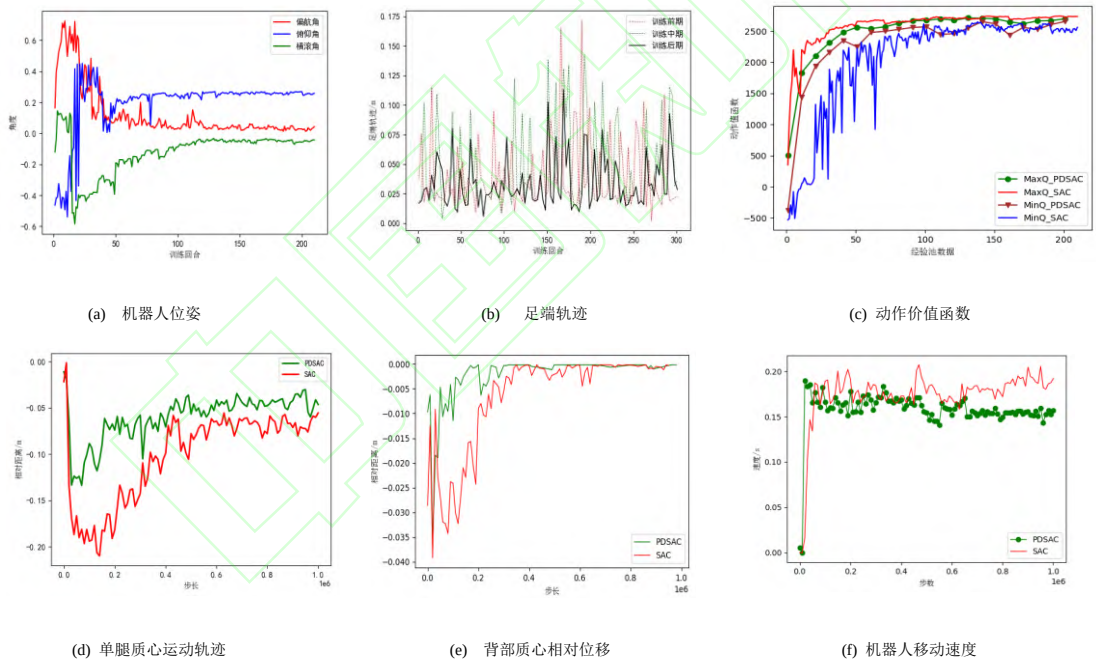
4.3 步态研究

引入 $\boldsymbol{\pi}_{\text{ce}}$ 后,为了防止评价网络 Q 值幅度相差过大,导致智能体进入到无用的状态空间,通过添加两个策略分布的距离作为约束条件,限制策略网络的更新幅度。

如式 19，表示 $\boldsymbol{\pi}_{\text{ce}}$ 在 $\boldsymbol{\pi}$ 的重要性采样率,θ 为评价网络参数,式 20 为相对熵公式,式中 ε 为重要性参数,通过 clip 函数限制重要性采样率的范围。通过 KL 散度和重要性采样率限制相对熵的 Q 值,从而降低策略蒸馏之间的计算复杂度,使混合策略选择动作值更大的策略,增加算法的可扩展性与稳定性。智能体在每一个状态都有确定的动作对,作为训练样本存入经验池中，利用高斯采样选取动作进入下个状态，并更新网络参数 θ，直至奖励收敛。

$$A_t(\theta)=\frac{\boldsymbol{\pi}_{\text{ce}}(\boldsymbol{a}_t\mid\boldsymbol{s}_t)}{\boldsymbol{\pi}(\boldsymbol{a}_t\mid\boldsymbol{s}_t)} \tag{19}$$

$$Q_{\text{LE}}(\boldsymbol{s})=\min \mathrm{E}_t\left[\sum_{i=1}^{n}\mathrm{KL}(\boldsymbol{\pi},\boldsymbol{\pi}_{\text{ce}}),clip(1-\varepsilon,1+\varepsilon)A_t(\theta_i)\right] \tag{20}$$



(a) 机器人位姿　(b) 足端轨迹　(c) 动作价值函数

(d) 单腿质心运动轨迹　(e) 背部质心相对位移　(f) 机器人移动速度

图 11　机器人在 PDSAC 算法的步态分析
Fig.11 Robot gait analysis in the PDSAC algorithm

$\boldsymbol{\pi}_{\text{ce}}$ 通过式(20)的相对熵限制状态价值函数过估计。如图 11,为 PDSAC 算法在仿真实验的表现:(a)为四足机器人身体姿态的变化。在训练前期,不论是俯仰角度，左右和上下摆动，浮动范围都较大,在训练后期，逐渐稳定，最终稳步向前。(b)为四足机器人在运动过程中足端轨迹变化,在训练前期,轨迹在不规则的上下摆动;训练中期,可以在一定的范围实现稳定的运动,说明机器人已经有更多的采样数据实现策略的更新;训练后期,摆动范围更小,机器人学习基本完成,可以快速的完成每次训练,更新采样动作。(c)为动作价值函数值,为了说明相对熵的作用,与 SAC 算法网络的 Q 值进行对比，PDSAC 算法只是将 SAC 算法的动作价值函数的最大值和最小值小幅度提高,不会造成 Q 值过估计。并且在收敛后,两

者的 Q 值高度重合,证明 PDSAC 算法只是在前期通过小幅度提高 Q 值加快探索效率,不会影响系统最终的稳定性。(d) 为单腿质心运动轨迹,反映四足机器人动态稳定性。在前期,评价网络和智能体都在进行无规则探索，PDSAC 和 SAC 算法都不能达到平稳步态。但由于 $\boldsymbol{\pi}_{ce}$ 的加入,使智能体更快的完成探索,到达平稳步态。(e)是背部质心的相对位移,表示机器人在行走中躯体的摆动程度。横坐标为训练步长,纵坐标为质心相对背部的距离。算法在前期有更多的动作样本数据,所以浮动程度小,并更快的平稳运动。(f)展示了机器人移动速度,横轴为训练步数,纵轴为每秒相对 x 轴前进速度, SAC 算法在平稳后速度还会有较大摆动,而 PDSAC 算法的速度则持续保持稳定,以牺牲速度换取步态的平稳。

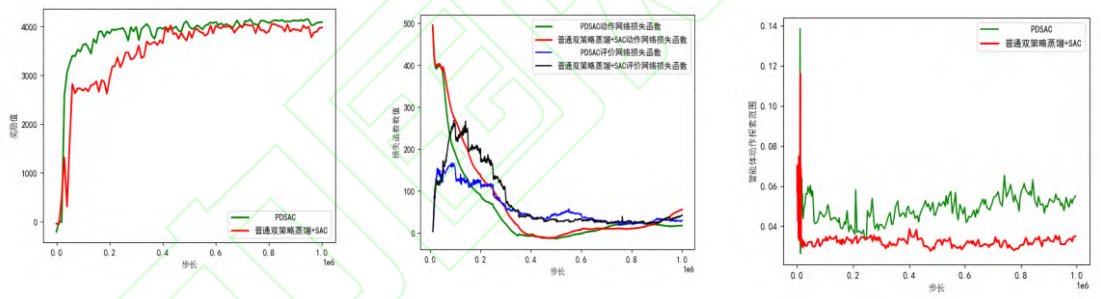
4.4 消融实验

为了验证本文提出的好奇心探索策略的优越性，本节设计对比试验。将普通双策略蒸馏框架结合 SAC 算法与 PDSAC 算法进行比较。普通双策略蒸馏使用同一环境中两个网络产生的策略相互策略形成组合策略,而 PDSAC 算法通过相对位移引入 $\boldsymbol{\pi}_{ce}$ 与 $\boldsymbol{\pi}$ 相互蒸馏形成混合策略。SAC 算法的相关参数如表 2 所示。

表 2 SAC算法参数

Table 2 SAC Algorithm Parameters

参数	数值
动作网络学习率	0.0003
评价网络学习率	0.0003
SAC 熵值正则化系数/ α	0.2
奖励折扣因子/ γ	0.99
每回合 Q 值衰减系数/ ε	0.99



(a) 奖励函数 (b) 网络损失函数 (c) 智能体动作探索范围

图 12 不同策略蒸馏框架对比

Fig.12 Comparison of different policy distillation frameworks

如图 12 所示：(a)表示普通蒸馏和 PDSAC 算法的奖励函数对比,PDSAC 中 $\boldsymbol{\pi}_{ce}$ 使奖励在前期增长较快,说明智能体在前期探索更多有利动作,策略更新加快,使奖励函数加速收敛。（b）为 Actor 网络和 Critic 网络参数更新过程中损失函数大小,随着步长的增加,损失函数都趋于零。由于 PDSAC 中 $\boldsymbol{\pi}_{ce}$ 使经验池的数据有更好的状态信息,评价网络的损失相比普通蒸馏更小,使智能体有更好的动作探索效率,动作网络更新加快,智能体更快的到达奖励收敛值。(c)展示了智能体在进行动作采样时策略的范围,可以看出,由于 $\boldsymbol{\pi}_{ce}$ 的作用,智能体的探索范围比普通蒸馏大,智能体可以更快的搜集环境状态信息使评价网络策略的更新加快,机器人更快的学习到稳定步态。在 10 万步时,两者出现明显差异,混合策略通过式(12)选择价值函数较大值,采样范围增加,采样动作更多,策略更新也越快。

5 结论

本文将策略蒸馏思想和 SAC 算法相融合提出一种 PDSAC 算法。该算法：

(1)利用原策略 Q 值生成 CE 策略结合原策略形成混合策略,通过状态值函数选择混合策略中更适合智能体探索的策略。通过 PD 的方法,使混合策略归一化并传输给 AC 网络框架。

(2)理论证明添加 CE 策略可以提高探索效率;通过 DML 使 CE 策略和原策略相互蒸馏,通过 KLD 限制相对距离使混合策略既有探索能力也有鲁棒性，并给出 PDSAC 的网络框架图。

(3)进行了仿真对比试验和消融实验,证明本算法在奖励函数收敛和稳定性方面较 SAC 算法和普通蒸馏结合 SAC 算法都具有优势。

未来,可以考虑使用此算法完成更多更复杂的四足机器人步态学习实验,也可以讨论是否可以将策略蒸馏思想应于除了 SAC 之外的其他强化学习算法。

参考文献 (References)

[1] LEVINE S, FINN C, DARRELL T, et al. End-to-end training of deep visuomotor policies[J]. The Journal of Machine Learning Research, 2016, 17(1):1334-1373.

[2] KOHL N, STONE P. Policy gradient reinforcement learning for fast quadrupedal locomotion[C]//Proceeding of IEEE International Conference on Robotics and Automation. New Orleans, USA:IEEE Press,2004: 2619-2624.

[3] LEE J, HWANGBO J, WELLHAUSEN L, et al. Learning quadrupedal locomotion over challenging terrain[J]. Science Robotics, 2020, 5(47):eabc5986.

[4] HWANGBO J, LEE J, DOSOVITSKIY A, et al. Learning agile and dynamic motor skills for legged robots[J]. Science Robotics, 2019, 4(26):eaau5872.

[5] HAARNOJA T, HA S, ZHOU A, et al. Learning to walk via deep reinforcement learning[J]. arXiv preprint arXiv:1812.11103, 2018.

[6] CZARNECKI W M, PASCANU R, OSINDERO S, et al. Distilling policy distillation[C]//Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics. Okinawa, Japan:PMLR,2019: 1331-1340.

[7] HONG Z W, NaAGARAJAN P, MAEDA G. Periodic intra-ensemble knowledge distillation for reinforcement learning[C]//Proceedings of Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021.Bilbao, Spain: Springer International Publishing, 2021: 87-103.

[8] AMIK F R, TASIN A I, AHMED S, et al. Dynamic Rectification Knowledge Distillation[J]. arXiv preprint arXiv:2201.11319, 2022.

[9] RUSU A A, COLMENAREJO S G , GULCEHRE C, et al. Policy distillation[J]. arXiv preprint arXiv:1511.06295, 2015.

[10] LAI K H, ZHA D, LI Y, et al. Dual policy distillation[J]. arXiv preprint arXiv:2006.04061, 2020.

[11] ZHAO C, HOSPEDALES T. Robust domain randomised reinforcement learning through peer-to-peer distillation[C]//Proceeding of the Asian Conference on Machine Learning,PMLR,2021: 1237-1252.

[12] 徐平安，刘全． 基于相似度约束的双策略蒸馏深度强化学习方法．[J] 计算机科学． 2022，11（12）：1-13．
XU P A, LIU Q, A dual-policy distillation deep reinforcement learning method based on similarity constraint[J]. Computer Science ,2022,11(12):1-13(in Chinese).

[13] LEE Y H, LEE Y H, LEE H, et al. Development of a quadruped robot system with torque-controllable modular actuator unit[J]. IEEE Transactions on Industrial Electronics, 2020, 68(8): 7263-7273.

[14] BURDA Y, EDWARDS H, STORKEY A, et al. Exploration by random network distillation[J]. arXiv preprint arXiv:1810.12894, 2018.

[15] LIU I J, PENG J, SCHWING A G. Knowledge flow: Improve upon your teachers[J]. arXiv preprint arXiv:1904.05878, 2019.

[16] LIU D, HAN H, SHEN F. Dialogue Policy Optimization Based on KL-GAN-A2C Model[C]//Proceedings of the International Computer Conference on Wavelet Active Media Technology and Information Processing. Chengdu, China, IEEE press ,2019: 417-420

[17] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.

[18] YANG C, YUAN K, ZHU Q, et al. Multi-expert l earning of adaptive legged locomotion[J]. Science Robotics, 2020, 5(49):eabb2174.

[19] WILLIAMS R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Machine Learning, 1992, 8(3): 229-256.

[20] HAARNOJA T, ZHOU A, ABBEEL P, et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor[C]//Proceedings of the International Conference on Machine Learning.Stockholm,Sweden:PMLR,2018:1861-1870.

[21] HAARNOJA T, TANG H, ABBEEL P, et al. Reinforcement learning with deep energy-based policies [C]//Proceedings of the International Conference on Machine Learning.Sydney,Australia,PMLR,2017: 1352-1361.

[22] HAARNOJA T, ZHOU A, HARTIKAINEN K, et al. Soft actor-critic algorithms and applications[J]. arXiv preprint arXiv:1812.05905, 2018.

[23] FUJIMOTO S, HOOF H V, MEGER D. Addressing function approximation error in actor-critic methods[C]//Proceeding of the International Conference on Machine Learning. Stockholm, Sweden: PMLR , 2018: 1587-1596.

[24] 王君逸， 王志， 李华雄， 等． 基于自适应噪声的最大熵进化强化学习方法[J]． 自动化学报， 2023， 49（1）： 54-66．
WANG J Y, WANG Z, LI H X, et al. A maximum entropy evolutionary reinforcement learning method based on adaptive noise[J]. Acta Automatica Sinica , 2023, 49(1): 54-66(in Chinese).

[25] MNIH V M, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning[C]// Proceeding of the International Conference on Machine Learning. New York, USA: PMLR,2016: 1928-1937.

[26] SHI H, ZHOU B, ZENG H, et al. Reinforcement learning with evolutionary trajectory generator: A general approach for quadrupedal locomotion[J]. IEEE Robotics and Automation Letters, 2022, 7(2): 3085-3092.

[27] AGARWAL R, LIANG C, SCHUURMANS D, et al. Learning to generalize from sparse and underspecified rewards[C]// Proceedings of the International Conference on Machine Learning. Long Beach,USA:PMLR,2019: 130-140.

[28] LI X, ZHANG X, NIU J, et al. A Stable Walking Strategy of Quadruped Robot Based on ZMP in Trotting gait[C]// Proceedings of the International Conference on Mechatronics and Automation. Wuhan, China:IEEE Press: 2022: 858-863.

[29] HINTON G, VINYALS O, DEAN J. Distilling the knowledge in a neural network[J]. arXiv preprint arXiv:1503.02531, 2015.

[30] CUI J, KINGSBURY B, RAMABHADRAN B, et al. Knowledge distillation across ensembles of multilingual models for low-resource languages[C]//Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).New Orleans,

USA:IEEE Press, 2017: 4825-4829.
[31] YING Z, TAO X, TIMOTHY M H, et al. Deep mutual learning[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA: IEEE Press ,2018: 4320-4328.
[32] WANG J, HU C, ZHU Y. CPG-Based Hierarchical Locomotion Control for Modular Quadrupedal Robots Using Deep Reinforcement Learning[J]. IEEE Robotics and Automation Letters, 2021, 6(4): 7193-7200.
[33] JOHNSON R W. An introduction to the bootstrap[J]. Teaching statistics, 2001, 23(2): 49-54.
[34] GIRI R .RAO B D . Bootstrapped sparse Bayesian learning for sparse signal recovery[C]//Proceedings of the Asilomar Conference on Signals, Systems and Computers. Pacific Grove, USA:IEEE Press,2014: 1657-1661.
[35] ZHANG Z, YAN J, KONG X, et al. Efficient motion planning based on kinodynamic model for quadruped robots following persons in confined spaces[J].IEEE/ASME Transactions on Mechatronics, 2021, 26(4): 1997-2006.