arXiv:2303.04356v2 [cs.LG] 2 Jul 2023

具有真正满足不等式约束的软演员-评论家算法

Taisuke Kobayashi，*IEEE* 会员

摘要—**强化学习中的软演员-评论家（SAC）有望成为下一代机器人控制方案之一。其最大化策略熵的能力有望使机器人控制器能够抵御噪声和扰动，从而适用于真实世界机器人应用。然而，当前实现会自动调整最大化策略熵的优先级，其规则可解释为等式约束，即将策略熵绑定在指定下界上。因此，当前的 SAC 不再最大化策略熵，这与我们的预期相悖。为解决 SAC 中的这一问题，本文改进了其实现，引入可学习的状态依赖松弛变量，以便通过将不等式约束重新表述为相应的等式约束，恰当地处理该约束并最大化策略熵。引入的松弛变量通过切换型损失函数进行优化，该函数兼顾满足等式约束和检验下界这两个目标。在 Mujoco 和 PyBullet 仿真器中，改进后的 SAC 在对抗攻击下的鲁棒性在统计意义上有所提升，同时对动作范数进行了正则化。本文还展示了一项真实机器人可变阻抗任务，以说明改进后的 SAC 在真实世界机器人控制中的适用性。尤其是，改进后的 SAC 保持了自适应行为，以应对训练期间从未经历过的物理人机交互。**

索引词—**强化学习、不等式约束、可变阻抗控制、物理人机交互。**

I. 引言

FOR 为了完成日益复杂的机器人任务（例如可变形物体操作 [1] 和人机交互 [2]），数据驱动控制，尤其是（深度）强化学习（RL） [3],，有望成为下一代控制方案。虽然学习此前仅限于游戏环境和动力学仿真器，但近年来已开始应用于真实世界机器人控制，并展现出良好的成果 [4]–[6]。

这一成功的关键因素之一，是最先进的学习算法——软演员-评论家（SAC） [7]。顾名思义，SAC 是一种演员-评论家算法，适用于机器人控制，因为它可以通过直接优化随机策略来生成实数形式的控制指令（即强化学习中的动作），例如关节力矩。SAC 通过在强化学习的主要目标中加入策略熵最大化，革新了学习机制；这有助于探索，同时抑制过度学习，并提升学习质量和速度。此外，作为一种在学习过程中，策略熵最大化还带来一个副作用：策略能够在各种情境中获得更强的错误恢复能力，而这些情境会在受到鼓励的探索过程中遇到 [8], [9]。这种能力体现为对噪声和扰动的鲁棒性；与控制质量一样，这也是真实世界机器人控制所必需的。此外，还有许多以 SAC 为基线提出的改进方法，例如：使 SAC 适用于离散动作空间 [10]; 结合归一化流改进策略模型 [11]；考虑最坏情况以确保安全 [12]。另外，研究者还在控制即推理的框架下重新推导并分析了 SAC [13]。

改进 SAC 的一个方向是自动调节温度参数。该参数决定策略熵最大化的优先级，并作为辅助目标加入 SAC。这一功能免去了针对不同任务进行手动调参的需要。最具代表性的自动调节方法由最初提出 SAC 的作者开发 [8]。通过将 SAC 重新解释为约束优化问题，温度参数被赋予拉格朗日乘子的含义，因此可将相应的优化过程应用于温度参数。主流库已实现带有此功能的 SAC [14], [15], ，并通过实验证明其效果简单而有效，不过其问题将在后文提出。其他方法也得到研究，例如使用元梯度 [16] ，以及引入好奇心以进一步加快探索 [17], ，但这些方法带来的改进似乎有限。

因此，SAC 已成为现代强化学习算法的标准，但如上所述，它仍在不断改进之中。本研究对温度参数的自动调节进行了修改。具体而言，本文首先指出，当前的标准实现可视为对策略熵施加等式约束后推导出的结果，这会导致策略熵收敛至其下界，而未能使其最大化 [18]。换言之，当前温度参数的自动调节会使 SAC 远未达到我们最大化策略熵的预期。如果 SAC 中的策略熵只是保持恒定，那么就可以像 DDPG [19] 和 TD3 [20], 那样在设计阶段将其固定，此时策略熵正则化可能收益甚微。除非恰当地纳入 SAC 理应具备的能力，即灵活调节高于下界的策略熵，否则就难以真正体现 SAC 的优势。

Taisuke Kobayashi 任职于日本东京千代田区一桥 2-1-2、邮编 101-8430 的国立信息学研究所（NII）和综合研究大学院大学（SOKENDAI）。稿件于2021年4月19日收到；于2021年8月16日修订。

0000–0000/00$00.00 © 2021年 IEEE

为解决这一问题，本文针对不等式约束 [21] 引入并优化一种可学习的状态依赖松弛变量，以最大化策略熵，使其能够适当地高于指定下界。只需将松弛变量加到下界上，传统自动调节便可正确满足策略熵的不等式约束。此外，由于松弛变量被近似为一个以状态为输入的函数，并采用深度神经网络实现，因此可以通过最小化反映其要求的损失函数（即适当地转换为等式约束并验证下界）轻松地对其进行优化。针对这类多目标优化，本文基于 ϵ无关损失 [22]，引入切换型标量化方法，而非简单的加权和。预计这一改进能够恰当地实现策略熵最大化，同时提升鲁棒性并缓解过度学习，从而在实际应用中更加稳定。

所提方法在 Mu-joco [23]和 Pybullet [24] 仿真器上进行了统计验证。结果表明，采用使用松弛变量的自动调节的改进 SAC 会调节（主要是增加）策略熵。这种熵最大化策略在对抗情境下具有更高的鲁棒性，同时对动作范数进行保守正则化。此外，通过展示真实机器人上的可变阻抗任务，说明了所提方法应用于真实世界机器人控制的适用性。即使对于训练期间完全没有经历过的物理人机交互，改进 SAC 也表现出了自适应行为。

II. 预备知识

A. 基本问题陈述

强化学习 [3] 解决的是最大化未来奖励之和（即回报）的最优控制问题。为此，假设存在由元组 ($\mathcal{S}$、 $\mathcal{A}$、 p_e、 r) 表示的马尔可夫决策过程（ MDP ），其中 $\mathcal{S}$ 和 $\mathcal{A}$ 分别表示状态空间和动作空间， p_e 给出状态转移概率， r定义当前情境下的奖励。具体而言，在未知环境的当前状态 $s \in \mathcal{S}$ 下，智能体根据其可训练策略 $\pi(a \mid s)$ 决定动作 $a \in \mathcal{A}$ ，并通过 a 与环境进行交互。结果是，根据状态转移概率 $p_e(s' \mid s、 a)$， s 被更新为下一个状态 s' ，同时获得奖励 $r = r(s、 a)$。

强化学习优化 π ，以通过从当前时间步 s 重复进行的交互随机最大化所获得的回报 $t、 R_t = \sum_{k=0}^{\infty} \gamma^k r($ s_{t+k}， $a_{t+k})$，其中 $\gamma \in [0, 1$为折扣因子。对于这一优化问题，强化学习通常定义以下两类价值函数，用于表示 R_t 的期望值。

$$V(s) = \mathbb{E}_{\rho^\pi}[R_t \mid s],\ Q(s,a) = \mathbb{E}_{\rho^\pi}[R_t \mid s, a] \tag{1}$$

其中 ρ^π 表示在 π （以及 p_e）条件下生成状态-动作轨迹的概率。请注意，在这种标准情况下， $V(s) = \mathbb{E}_{a\sim\pi}[Q(s,$ $a)]$成立。人们提出了各种学习算法来学习这些函数以及 π，下文将介绍的 SAC 就是其中之一。请注意，为了学习 π、V 和/或 Q，人们广泛使用深度神经网络作为它们的函数逼近器。本文也采用了这种流行的实现方式。

B. 软演员-评论家：SAC

下面介绍 SAC。近年来，它在强化学习领域取得了巨大成功，并已成为一种标准方法 [7]。它的主要特点是通过将策略熵$\mathcal{H}(\pi) = \mathbb{E}_\pi[-\ln \pi]$ 加入奖励中，来求解熵最大化问题，如下所示：

$$r = r(s,a) + \alpha \mathcal{H}(\pi(\cdot \mid s)) \tag{2}$$

其中， $\alpha \geq 0$ 表示用于调整策略熵优先级的温度参数。

根据这一扩展， V 与Q之间的关系首先给出如下：

$$V(s) = \mathbb{E}_{a\sim\pi}[Q(s,a) - \alpha \ln \pi(a \mid s)] \tag{3}$$

由此，软贝尔曼方程为 Q 推导出以下自举监督信号，即 y。

$$\begin{aligned} &Q(s,a) \\ &= r(s,a) + \gamma \mathbb{E}_{p_e(s'|s,a)}[V(s')] \\ &= r(s,a) + \gamma \mathbb{E}_{p_e(s'|s,a),\pi(a'|s')}[Q(s',a') - \alpha \ln \pi(a' \mid s')] \\ &= y \end{aligned} \tag{4}$$

众所周知，通过优化近似的 Q 以满足这一等式，可以正确地将 Q 作为回报的期望值获得。

在 SAC 的实际实现中，会准备两个不同的函数逼近器来逼近 $Q_{1,2}$，用于避免对 Q 的过高估计。此外，还会提供与它们对应的 ¯目标网络 $Q_{1,2}$ [25]，以计算 y。因此，通过求解以下时间差分（TD）误差最小化问题，即可完成对 $Q_{1,2}$ 的学习。

$$\begin{aligned} Q_{1,2}^* &= \arg\min_{Q_{1,2}} \mathbb{E}_{(s,a,s',r)\sim\mathcal{D},a'\sim\pi}\left[\frac{1}{2}(y - Q_{1,2}(s,a))^2\right] \\ y &= r + \gamma\Big\{\min_{i=1,2} \bar{Q}_i(s',a') - \alpha \ln \pi(a' \mid s')\Big\} \end{aligned} \tag{5}$$

其中， $\mathcal{D} = \{(s,a,s',r)_i\}_{i=1}^N$ 表示容量为N的回放缓冲区，其中以元组 (s,a,s',r)的形式存储马尔可夫决策过程上的交互数据。

利用训练好的 Q，可以按如下方式优化 π ，使其在任意状态下最大化 V （或最小化 $-V$）：

$$\begin{aligned} \pi^* &= \arg\min_\pi \mathbb{E}_{s\sim\mathcal{D}}[-V(s)] \\ &= \arg\min_\pi \mathbb{E}_{s\sim\mathcal{D},a\sim\pi}\Big[-\min_{i=1,2} Q_i(s,a) + \alpha \ln \pi(a \mid s)\Big] \end{aligned} \tag{6}$$

在这里，通过最大化较小的 Q，预期可以实现保守更新。该最小化问题可以利用重参数化技巧求解 $a \sim \pi$ [26]。也就是说，对于 π ，直接计算上述损失函数的梯度（更具体地说，是其参数的梯度，例如高斯分布中的位置参数和尺度参数），以进行基于梯度的优化。

最后，本文展示了如何自动调节 α。迄今为止， α一直被视为一个常数参数，而针对每个任务对其进行微调至关重要，但耗时较长。这是由 SAC [8] 的同一批作者在文献中提出的新定义。具体而言，

首先，将 SAC 的优化问题重新表述为一个关于策略熵的约束优化问题。

$$\pi^* = \arg\min_{\pi} \mathbb{E}_{s\sim\mathcal{D},\rho^\pi}\left[-\sum_{k=0}^{\infty}\gamma^k r(s_{t+k}, a_{t+k}) \mid s\right] \quad (7)$$
$$\text{s.t.} \quad \mathcal{H}(\pi(\cdot \mid s)) \geq \mathcal{H}^* \ (^\forall s \in \mathcal{D})$$

其中，$\mathcal{H}^*$ 表示下界，对于连续动作空间，通常将其设定为 $-|\mathcal{A}|$。

在这种情况下，恢复为常规的 SAC 优化问题时，α 即为拉格朗日乘子。基于这一新的解释，α 可以通过以下最小化问题进行优化。

$$\alpha^* = \arg\min_{\alpha} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi}[-\alpha\{\ln\pi(a \mid s) + \mathcal{H}^*\}] \quad (8)$$

理论上，此问题中的 π 应为 π^*，但由于未明确给出π^*，实际中以 π 代替。因此，从定性角度看，如果 $\ln\pi > \mathcal{H}^*$，则会增大 α 以最大化熵；否则，如果 $\ln\pi < \mathcal{H}^*$，则会减小 α 以允许更低的熵。

˜请注意，为了满足 $\alpha \geq 0$，将 $\tilde{\alpha} \in \mathbb{R}$（例如 $\alpha = e^{\alpha}$）设为可训练参数，并使用镜像下降法 [27]对其进行更新。也就是说，使用式（8）关于 α的梯度来更新 $\tilde{\alpha}$，$\mathbb{E}_{s\sim\mathcal{D},a\sim\pi}[-\{\ln\pi(a \mid s) + \mathcal{H}^*\}]$，然后再次将其转换为 α。这样可以避免由于满足 $\alpha \geq 0$的非线性函数而导致 $\alpha \simeq 0$ 中的梯度消失问题。

III. 提案

A. SAC中的未解决问题

据报道，SAC 的控制性能（例如盈利能力和鲁棒性）优于以往的强化学习算法。这主要归因于在不失去原有目的（即奖励）的前提下，尽可能最大化策略熵 $\mathcal{H}(\pi)$。然而，上文介绍的实际实现并不一定能使 $\mathcal{H}(\pi)$ 达到最大值。

这是因为，如式（8）所定义的自动调节温度参数 α，只会使 $\mathcal{H}(\pi)$（或 $-\ln\pi$）趋近于 $\mathcal{H}^*$，这符合其定性行为的预期（文献 [18] 中也有相关报告）。换言之，上述自动调节被认为是基于带有等式 约束的拉格朗日乘子法推导而来，这与最大化 $\mathcal{H}(\pi)$ 不一致。因此，尽管优化问题最初旨在通过不等式 约束最大化 $\mathcal{H}(\pi)$，但其实现采用的却是 $\mathcal{H}(\pi) = \mathcal{H}^*$。由于这种行为，$\mathcal{H}^*$ 常被称为“目标”熵，尽管它应被定义为下界。$\alpha \geq 0$ 的局限性可能导致 $\mathcal{H}(\pi) > \mathcal{H}^*$，而无法完全满足这一等式约束，但在实践中，当前的自动调节使得

$\mathcal{H}(\pi)$ 在训练过程中大多收敛到 $\mathcal{H}^*$（详见后文的仿真结果）。因此，若没有合适的 $\mathcal{H}^*$，策略熵预计会过低，无法达到预期性能，也可能无法充分发挥最大化策略熵所带来的全部益处。

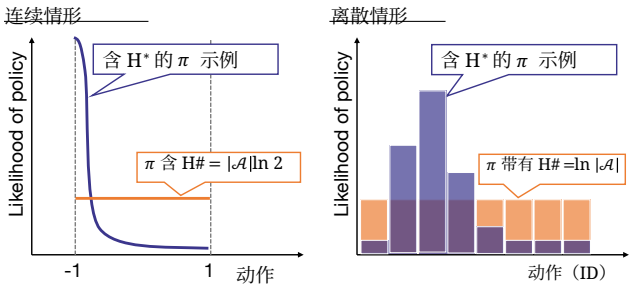


图 1：策略熵的上界

B. 带松弛变量的自动调节温度

为了解决这一问题，本研究采用另一种方式，将棘手的不等式约束转换为简单的等式约束。这样就可以像对 α 进行传统自动调节时那样，使用拉格朗日乘子法处理转换后的等式约束。为此，通常会引入松弛变量 $\Delta \geq 0$ [21]（即使在近期的机器学习 [28]中也是如此）。

$$\begin{aligned} &\mathcal{H}(\pi(\cdot \mid s)) \geq \mathcal{H}^* \quad (^\forall s \in \mathcal{D}) \\ \Leftrightarrow\ &\mathcal{H}(\pi(\cdot \mid s)) = \mathcal{H}^* + \Delta(s) \quad (^\forall s \in \mathcal{D},\ \Delta \geq 0) \end{aligned} \quad (9)$$

简而言之，通过求得一个在任意状态下都满足 $\Delta(\mathcal{H})\pi - \mathcal{H}^* = \Delta$ 的量，理论上可以将不等式约束转化为等式约束。

给定适当的 Δ，α 的优化可由式 (8) 改写为以下形式。

$$\alpha^* = \arg\min_{\alpha} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi}[-\alpha\{\ln\pi(a \mid s) + \mathcal{H}^* + \Delta(s)\}] \quad (10)$$

作为对这一更新规则的一种简单解释，α 使 $\mathcal{H}(\pi)$（或 $-\ln\pi$）收敛到 $\mathcal{H}^* + \Delta \geq \mathcal{H}^*$，后者是新设定的目标值（此前，$\mathcal{H}^*$ 承担这一角色，而不是策略熵的下界）。此外，由于 Δ 被视为状态相关函数，因此可以预期不同状态下的策略熵各不相同：例如，对于需要准确性的场景，策略熵可能较小；对于需要探索的场景，则可能较大。

C. 松弛变量的设计

Δ 专门设计为状态相关函数。它由以状态为输入的深度神经网络近似。由于深度神经网络的输出 d是实数值，因此需要将其映射到 Δ 的定义域。在一般的松弛变量中，Δ 被定义为非负值。此外，在 SAC 中还可以确定其上界。

具体而言，SAC 主要针对有界连续动作空间，这种空间在许多任务中十分常见，并通过显式变换动作空间使得 $\mathcal{A} = [-1, 1]^{|\mathcal{A}|}$。在这样的盒约束空间中，策略可以服从均匀分布，$\pi = 2^{-|\mathcal{A}|}$，其最大熵显而易见（即 $\mathcal{H} = |\mathcal{A}|\ln 2$，如图 1 左侧所示）。类似地，即使在离散动作空间中，均匀分布 $\pi = |\mathcal{A}|^{-1}$ 也能使熵最大化（即 $\mathcal{H} = \ln|\mathcal{A}|$，如图 1 右侧所示）。

总之，SAC 中策略熵的上界 Δ Δ 可表示如下：

$$\overline{\Delta} = \begin{cases} |\mathcal{A}| \ln 2 - \mathcal{H}^* & \text{(Continuous)} \\ \ln |\mathcal{A}| - \mathcal{H}^* & \text{(Discrete)} \end{cases} \tag{11}$$

其中，原始论文建议连续动作空间使用 $\mathcal{H}^* \ -|\mathcal{A}|$ [8]，而离散动作空间在 SAC-Discrete 中调整为 0.98 $\ln |\mathcal{A}|$ [10]，，尽管这两个值针对的是传统的等式约束。请注意，必须给定 $\mathcal{H}^*$，使得 $\Delta \gtrsim 0$。尽管目前推荐的值满足这一条件，但仍希望找到新的推荐值，因为此前的 $\mathcal{H}^*$ 并未发挥下界的作用。

结合 Δ 以及 $\Delta \geq 0$这一事实，对应网络的实值输出 d 可以通过 Sigmoid 函数映射到其定义域内的 Δ，并与 Δ 相乘。也就是说，d 通过以下公式转换为 $\Delta \in [0, \Delta]$ 。

$$\Delta = \overline{\Delta}\sigma(d) \tag{12}$$

其中，$\sigma(\cdot)$ 表示 Sigmoid 函数。请注意，尽管$\sigma(d) = 1/(1+\exp(-d))$ 是最具代表性的设计，本文采用了文献 [29], [30], 中的 squareplus 函数的一阶导数，该函数具有重尾特性，并且在边界（即 0 和 Δ）附近具有更高的数值分辨率。

D. 松弛变量优化

所设计的 $\Delta \in [0, \overline{\Delta}]$ 通过最小化适当的损失函数进行优化。通常，拉格朗日函数（即式（6）中针对 π 的损失函数）也应相对于 Δ进行最小化。换言之，应最小化 $\alpha\Delta$ 以消除 Δ，其速度取决于α，并用于检查边界处（即下界）的性能。具体而言，当 α 足够小时，可以允许$\mathcal{H}(\pi)$ 减小，此时 Δ 的减小速度将放缓，以适度降低 $\mathcal{H}(\pi)$。

另一项要求是最小化式（9）左右两侧的误差，以满足转换后的等式约束。只需最小化式（9）左右两侧的绝对误差即可实现这一点。然而，如果仅通过优化 Δ 就能充分满足等式约束，那么 α 将不再有自动调节的空间。为避免这一问题，采用 ϵ 不敏感损失函数 [22] 。也就是说，如果误差低于某个阈值 $\epsilon \geq 0$，就停止优化 Δ 以满足等式约束。

最后，需要在上述两个损失函数之间取得平衡。一个简单的解决方案是
加权和，但这需要引入并调整一个新的权重超参数。我们不采用加权和，而是考虑根据 ϵ 在两个损失函数之间进行切换的机制。也就是说，当等式约束未达到ϵ所规定的某个水平时，仅优化 Δ 以满足等式约束；否则，逐渐将 Δ 减小至零。

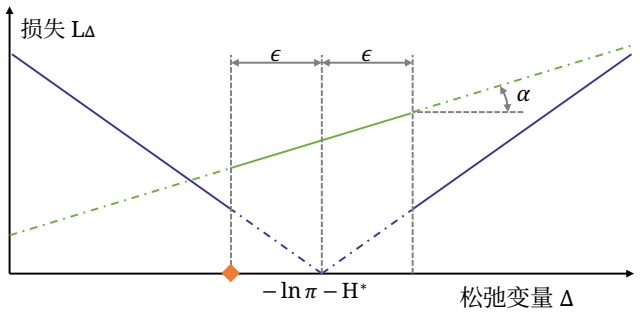


图 2：松弛变量 Δ 的损失函数

总之，通过以下最小化问题对 Δ 进行优化。

$$\Delta^* = \arg\min_{\Delta} \mathbb{E}_{s\sim\mathcal{D}, a\sim\pi}[\mathcal{L}_\Delta(s, a; \epsilon)] \tag{13}$$

$$\mathcal{L}_\Delta(s, a; \epsilon) = \begin{cases} |\ln \pi(a \mid s) + \mathcal{H}^* + \Delta(s)| & |e| > \epsilon \\ \alpha\Delta(s) & |e| \leq \epsilon \end{cases}$$

其中 $|e| = |\ln \pi(a \mid s) + \mathcal{H}^* + \Delta(s)|$。通过最小化该损失函数，如果 $\ln \pi$ 处于平稳状态，则会满足 $\ln \pi + \mathcal{H}^* + \Delta = -\epsilon$，如图 2 所示。然而在实践中，平衡点会根据自动调节的 α，因 $\ln \pi$ 而发生偏移，并相应地对 Δ 进行优化，这是交替优化 [31]的一个方面。

需要说明的是，在本文的实现中，上述最小化问题是在优化 α之后，采用镜像下降法求解的。也就是说，利用 $\mathcal{L}_\Delta$ 关于 Δ 的梯度来更新 d （即映射到 Δ之前的值）以及涉及 d的网络参数。这可以通过将式（13）重新定义为以下最小化问题来实现。

$$d^* = \arg\min_{d} \mathbb{E}_{s\sim\mathcal{D}, a\sim\pi}[\mathcal{L}_d(s, a; \epsilon)] \tag{14}$$

$$\mathcal{L}_d(s, a; \epsilon) = \begin{cases} \operatorname{sign}(\ln \pi(a \mid s) + \mathcal{H}^* + \Delta(s))d(s) & |e| > \epsilon \\ \alpha d(s) & |e| \leq \epsilon \end{cases}$$

其中，sign(·)表示符号函数。该实现可以缓解由 σ引起的梯度消失问题，因为它将 d 映射到 Δ。

IV. 仿真

A. 仿真条件

为对所提出的 SAC 改进进行统计验证，我们在 Mujoco [23] 和 Pybullet [24]上进行了动力学仿真。在这两种仿真器上均开展了以下三类任务：

- *Hopper*（Hopper-v4/HopperBulletEnv-v0）
- *HalfCheetah*（HalfCheetah-v4/HalfCheetahBulletEnv-v0）
- *Ant*（Ant-v4/AntBulletEnv-v0）

其中，每个括号中的第一个和第二个标签分别表示在 Mujoco 和 PyBullet 中实现的任务。为便于区分这两种仿真器，任务名称以“M”结尾表示 Mujoco，以“B”结尾表示 PyBullet。每项任务均训练 2000 回合。

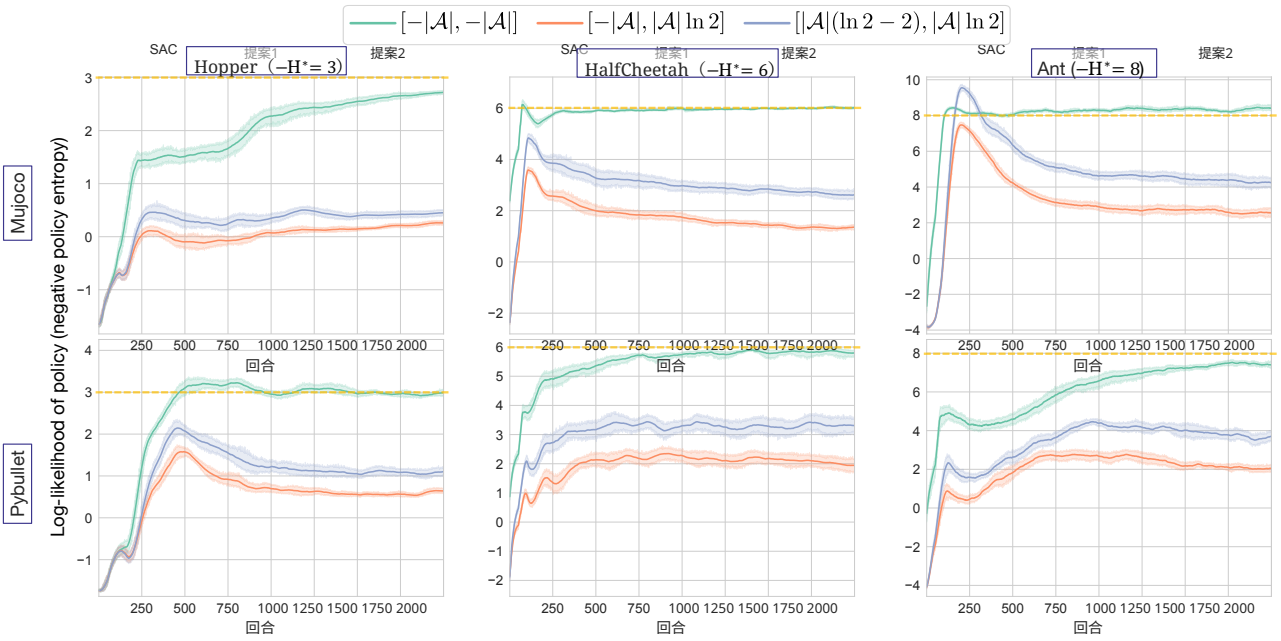


图3：策略对数似然 $\ln \pi \simeq -\mathcal{H}(\pi)$ 的学习曲线

比较以下三种条件：

- 传统情形：$\mathcal{H}^* = -|\mathcal{A}|$ 与 $\Delta = 0$（即 $\mathcal{H}(\pi) \in [-|\mathcal{A}|, -|\mathcal{A}|]$）
- 所提情形1：$\mathcal{H}^* = -|\mathcal{A}|$ 满足 $\Delta \in [0, \Delta]$（即 $\overline{\mathcal{H}(\pi)} \in [-|\mathcal{A}|, |\mathcal{A}| \ln 2]$）
- 所提情形2：$\mathcal{H}^* = \mathcal{H} - 2|\mathcal{A}|$ 满足 $\Delta \in [0, \Delta]$（即 $\overline{\mathcal{H}(\pi)} \in [|\mathcal{A}|(\ln 2 - 2), |\mathcal{A}| \ln 2]$）

第一种对应传统方法，第二种是采用与传统方法相同下界的所提方法 [8]，第三种则采用小于传统建议值的下界（即 $\mathcal{H}^* \simeq -1.3|\mathcal{A}|$）。通过比较第一种与其他方法，可以确认所提松弛变量能够恰当地满足不等式约束。通过比较第二种和第三种，可以定性分析指定下界所造成的性能差异。请注意，所有其他配置均相同，如附录A所述。

对于上述任务与方法的每种组合，均使用不同的随机种子训练了共 24 次。为了从统计学角度评估所获得策略的性能，我们在相应任务中对每个学习得到的策略进行了 100 次测试，并采用受文献 [32] 启发的对抗攻击。具体而言，以 5% 的概率（Hopper）和 20% 的概率（HalfCheetah 和 Ant）将动作替换为由高斯噪声生成的动作，并在 $[-0.2, 0.2]$ 范围内进行变换。我们考虑了以下两个鲁棒性评价指标：一是动作平均范数，用于判断所获得的策略行为是否保守；二是每次测试期间的奖励总和，用于体现所获得的策略是否维持了控制性能。

B. 学习趋势

策略对数似然 $\ln \pi$（对应于负的策略熵 $-\mathcal{H}(\pi)$）的回合平均值为如图3所示。此外，回报 R 的学习曲线如图4所示。请注意，这些曲线是使用来自优化策略 π 的采样动作获得的。

首先，结果表明，传统 SAC 中的 $\ln \pi$ 大致停留在 $|\mathcal{A}|$，且对 α 的优化满足了下界 $\mathcal{H}^* = -|\mathcal{A}|$ 的等式约束。另一方面，正如相应置信区间的宽度所示，所提方法平均而言明显降低了 $\ln \pi$，同时允许其出现更多波动。也就是说，所提方法为策略熵根据不同情况增加或减少留出了空间。所提方法不仅通过维持不等式约束实现了策略熵的这种适应性，还使策略熵（平均而言）保持在下界之上。需要注意的是，第三种情况的策略熵通常低于第二种情况，这可能是某种初始值依赖性所致，尽管也有可能只是较低的下界拉低了整体数值。也就是说，如果输出 Δ 的深度神经网络采用相同的随机种子进行初始化，那么初始输出差异与 Δ 成正比，因此，用于优化 Δ 的搜索起点可能会发生变化，并找到不同的平衡点。

自适应策略熵的优势在 HopperM 和 AntM 中得到了证实：通过抑制过度学习，R 相较于传统 SAC 有所提升。此外，在学习的早期阶段，HalfCheetahM 和 AntB 中也表现出了更高效的探索。另一方面，研究发现，较高的策略熵有时会因动作的确定性较低而妨碍性能提升。R，尤其是在 HalfCheetahM 中，随着所提方法的策略熵增加，实际上有所下降；而第三个条件（降低下界）通过防止策略熵变得过大，缓解了这一不利影响。尽管如此，对

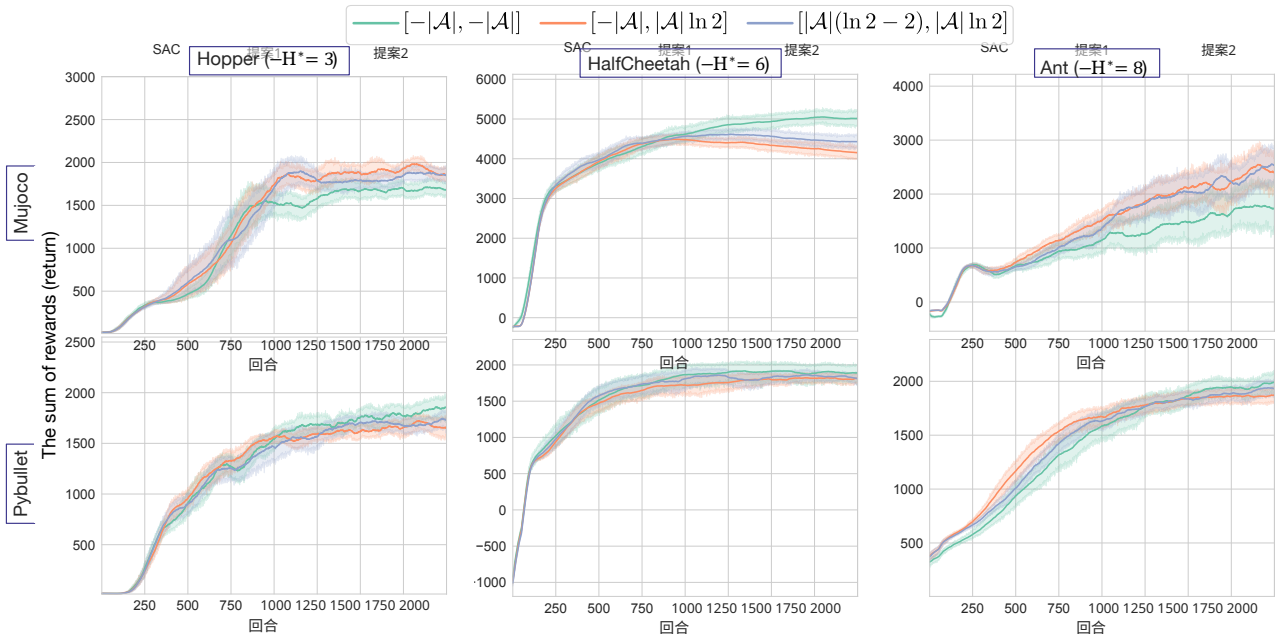


图4：回报的学习曲线 R

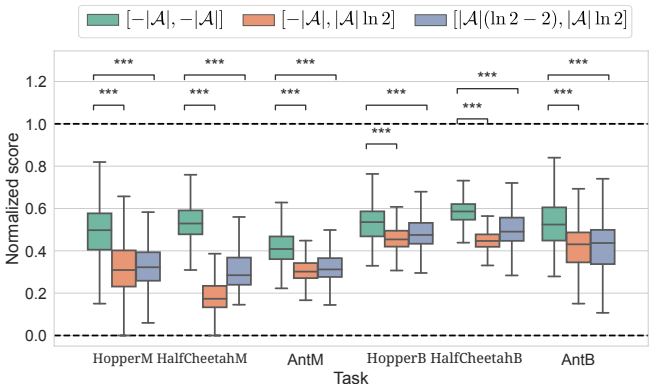


图5：扰动测试中的归一化动作范数

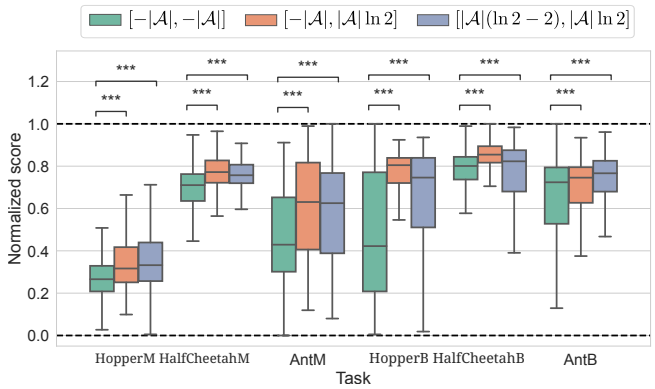


图6：扰动测试中的归一化回报

所提方法对 Δ 的优化施加了若干目标，其中之一是达到边界（即下界），但这一目标尚未完全实现。未来有必要设计更精细的损失函数。

C. 测试结果

通过对抗攻击下的测试结果，研究了各条件下所获得策略的统计性能（尤其是鲁棒性）。首先，对噪声和扰动具有鲁棒性的随机模型通常倾向于采取保守策略（例如，其输出接近于零） [33]。当像 SAC 那样通过熵最大化来生成有界随机值时，这一点可以得到更直接的解释。也就是说，由于边界附近的动作必然具有较高的概率，通过最大化策略熵来抑制这些动作，会迫使动作远离边界并趋向中心（即零）。从机器人控制的角度来看，这种保守性在无形中提高能源效率并使行为更加平稳，与相关文献中的做法类似 [34]。

图 5 展示了动作的 L2 范数的回合平均值。请注意，统计显著性通过 scipy.stats 包中的 mannwhitneyu(alternative="less") 得到确认，该检验用于确认第一种条件是否大于其他条件。结果一目了然：动作范数与策略熵呈负相关。换言之，与传统方法相比，成功提高策略熵的两种方法确实实现了更保守的行为。不过，两者并非简单地成反比。例如，在 AntM 和 AntB 中，尽管第二种和第三种条件下的策略熵相差较大，但动作范数的差异很小。这一现象符合预期，因为 AntM 和 AntB 中的最优行为会经历相变，此时策略熵可以高于下界，从而正确体现不等式约束。传统方法不仅会因策略熵最大化而失去



图7：可变阻抗任务中使用的机械臂

保守行为，还会忽略这样的最优解。

如果所获得的策略仅在行为上表现得保守，则不能称其具有鲁棒性。当智能体因对抗攻击偏离最优轨迹时，策略必须能够使其恢复。这种能力可以通过评估对抗攻击下的控制性能（即回报 R）来确认。由于采用熵最大化策略的智能体在使用采样动作进行训练的过程中，应当会遇到各种情况，并学习如何从这些情况中恢复。也就是说，正如文献 [8], [9]所指出的，预期熵最大化策略具有鲁棒性。

此处，图6对回报进行了统计描述。需要注意的是，使用 mannwhitneyu(alternative="greater") 确认统计显著性，该检验用于确认第一个条件是否小于其他条件。因此，尽管如图4所示，传统方法有时优于其他方法，但它容易受到对抗攻击的影响，在统计意义上表现得比所提方法更差。策略熵越大，控制性能就越具有鲁棒性，这一点可由控制性能总体得以保持这一事实得到印证。然而，第三个条件的策略熵较小，其在无对抗攻击时的初始控制性能高于第二个条件，并且 HopperM 和 AntB 的中位数、Q1 和 Q3 均更高。因此，具有较小下界的第三个条件能够更好地平衡策略熵，因而更值得选择，尤其是在通过一定改进使下界的验证函数得到良好执行的情况下。

D. 真实机器人上的演示

E. 设置

通过上述仿真可以确认，通过真正最大化策略熵，所提方法相比传统 SAC 对扰动具有更强的鲁棒性。下一步将展示所提方法在真实世界机器人控制中的实用性。本实验使用的机器人是图7所示的五自由度机械臂。在跟踪随机给定的机器人末端执行器轨迹时，使用强化学习优化六个阻抗参数

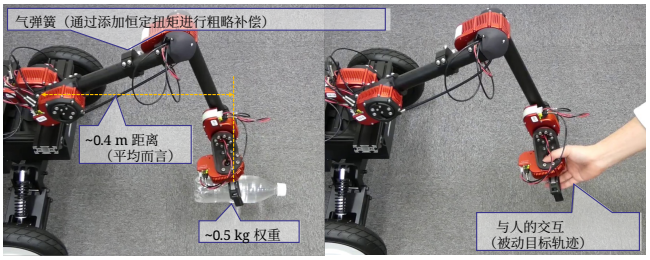


图 8：负载和交互情境

，构成可变阻抗控制系统 [35], [36]。该优化旨在将作用力降至最低，同时使轨迹误差保持在阈值以内。实验配置的详细信息总结于附录B中。

对每种方法使用不同的随机种子训练五个模型，并在训练期间或训练后，按照以下评价指标评估每个回合：回报、跟踪误差、超过阈值的频率、施加力、动作范数以及策略的对数似然。前四项是控制目标的评价指标，后两项是策略鲁棒性的评价指标。为了展示所提方法训练出的策略的鲁棒性，我们在三种情境（见图 8）中分别进行了 10 次测试。

- 原版：跟踪一条训练阶段未包含的随机轨迹。
- 负载：末端执行器抓取约 0.5 kg 的重物，同时调节肩部的偏置力矩。
- 交互：通过人与末端执行器之间的物理交互，人被动地移动末端执行器，从而设定末端执行器的目标位置；其范围则限制在训练期间经历过的范围内。

在*权重* 情境下，偏置力矩（见附录 B）会进行调整，以大致补偿重量，从而在肩部产生约 $0.5\,\mathrm{kg} \times -9.8\,\mathrm{m/s}^2 \times 0.4\,\mathrm{m} \simeq -2.0\,\mathrm{Nm}$ 的作用。这样可以避免末端执行器过度下降，防止出现重量频繁触地、无法重复的情况，同时水平动力学也必然会发生变化。在*交互* 情境下，通过正向运动学测量末端执行器的当前位置，并将其经过低通滤波器处理后设为目标位置。值得注意的是，为了展示训练带来的性能提升，也会测试阻抗参数固定为初始值的情况。

F. 学习趋势

评估指标的学习曲线如图 9 所示。此处，“SAC”表示传统 SAC，“提案”表示采用 $\mathcal{H}^* = \mathcal{H} - 2|\mathcal{A}|$ 的所提方法。尽管所提方法能够略微减小跟踪误差，但正如上述仿真中所证实的那样，在学习过程中两者的回报大多相近。另一方面，所提方法的策略熵明显高于传统方法（即策略的对数似然更小），这表明即使在真实机器人实验中，所提方法也能够正确处理不等式约束。相应地，动作范数更小，

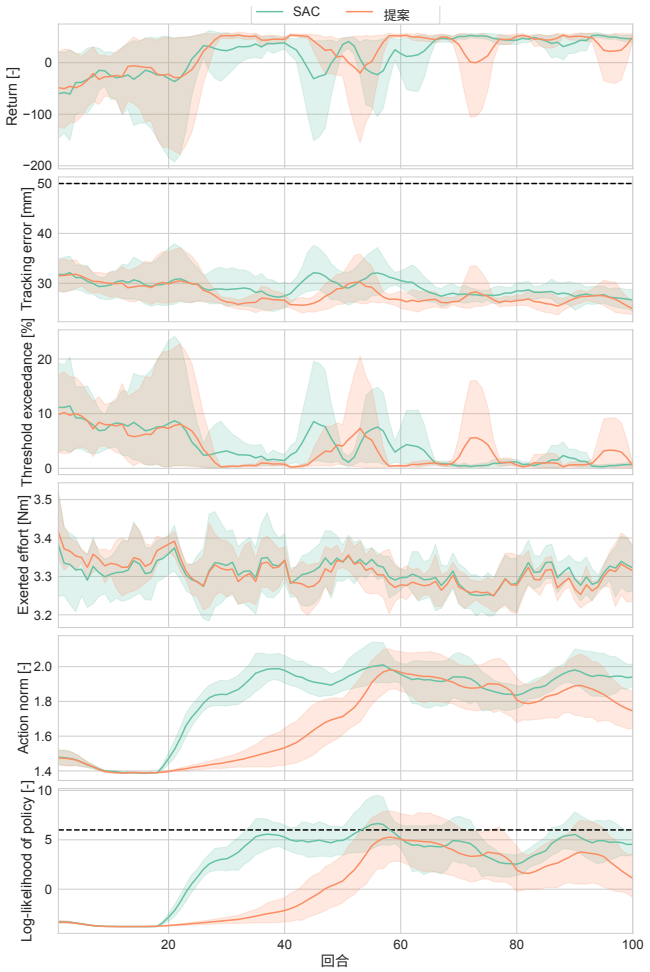


图 9：可变阻抗任务的学习曲线

使策略更加保守。由于该方法能够使阻抗参数平滑更新，因此其将在如下所述的该系统稳定运行中发挥重要作用。

G. 测试结果

表 I 总结了上述三个情境的结果，相关示例见随附视频。首先，在 Vanilla场景中，SAC 和提案的表现均优于固定方案。单看回报均值，SAC 和提案之间没有明显差异；但从标准差来看，提案显著提升了性能的稳定性。这是因为提案学会了在留有一定误差余量的同时不超过阈值，这一点体现在跟踪误差和阈值超限上。请注意，学习后施加力略有增加，这表明需要额外的力矩来提高跟踪准确性。

其次，在 Weight场景中，随着抓取重量导致动力学发生变化，性能不可避免地下降。在这种情况下，Fixed 达到了最佳性能，因为其他经过优化的阻抗参数偶尔会在 y 轴运动中引发共振。不过，与 SAC 相比，Proposal 引发共振的频率更低，并且保持了接近 Fixed 的性能（事实上，以中位数进行评估时，其性能优于 Fixed）。

最后，在 Interaction场景中，尽管施加了无效的力矩，Fixed 仍无法跟踪人体运动。相比之下，Proposal 能够保持相对良好的性能，并在所有评价指标上均取得最佳结果。特别是，Exerted effort 在其他情境中为提高跟踪准确性而被允许略微增加，但在本情境中，其增幅小于其他方法，且跟踪误差和阈值超限也均为最佳。这表明，Proposal 并未为了提高跟踪准确性而将阻抗参数设置得过于僵硬，而是根据具体情况进行了适当优化。换言之，Proposal 仅在跟踪误差较大且很可能超过阈值时才提高刚度；除此之外则保持柔顺，从而能够以较小的作用力灵活地跟踪人体对外部力的响应。

因此，结果证实，所提方法能够通过正确地最大化策略熵，使其策略即使面对现实世界扰动也具有鲁棒性。一个有趣的趋势是，在受到扰动的情境下，策略对数似然有所增加。这表明，所提方法能够恰当地应对未经历过的事件，因为它从相似经验中学习到了保守的策略；而传统 SAC 则因鲁棒性不足而产生了错误动作。

五、结论

本文探讨了 SAC 中温度参数的一种新的自动调节方法。具体而言，以往的方法不再最大化策略熵，而只是将其约束在下界（因此也称为目标熵）。这是因为这种自动调节方式被解释为从等式约束推导而来。为解决这一问题，本文使用状态相关松弛变量，将用于最大化策略熵的期望不等式约束恰当地转换为相应的等式约束。此外，对松弛变量进行优化，使其在满足等式约束的同时尽可能取较小值。通过所提改进，在 Mujoco 和 PyBullet 上进行的仿真验证了该方法在对动作范数进行正则化（从而使策略更加保守）的同时，能够提高对扰动的鲁棒性。最后，通过真实机器人实验展示了所提方法的实用性。即使在训练期间完全没有相关经验，所提方法在物理人机交互中也表现出了自适应行为。

然而，在不存在扰动的情况下，所提方法有时不如传统的SAC。应在近期研究一种用于调整下界 $\mathcal{H}^*$ 的框架（如文献 [18],中所述），或研究一个理论上更有效的下界。

附录

A. 实现细节

SAC 使用 PyTorch [37], 实现，基本实现遵循相关文献 [25], [34]。在每个回合中，通过使用随机采样的动作与环境交互，获得元组 $(s,\ a,\ s',\ r)$，并将其存入回放缓冲区 $\mathcal{D}$，缓冲区最大容量为 $N = 102,\ 400$。回合结束时，最多一半的经验会以均匀随机的方式重放，并将其拆分成最大容量为

表 I：可变阻抗任务的评价指标：每种情境下的最佳结果以粗体标出。

Test	方法	评价指标：均值（标准差）					
		回报 [–]	跟踪误差 [mm]	阈值超限 [%]	施加力 [Nm]	动作范数 [–]	策略对数似然 [–]
原版	固定	36.44 (14.28)	30.18 (1.77)	1.52 (1.32)	**3.36** (0.16)	N/A N/A	N/A N/A
	SAC	44.24 (20.51)	**25.21** (3.07)	0.70 (1.43)	3.40 (0.21)	1.84 (0.06)	5.61 (0.69)
	提案	**46.68** (14.75)	25.69 (2.85)	**0.56** (0.60)	3.39 (0.21)	**1.59** (0.16)	**2.71** (2.57)
权重	固定	**23.01** (20.57)	28.99 (1.91)	**1.26** (1.47)	**4.06** (0.39)	N/A N/A	N/A N/A
	SAC	12.28 (45.71)	28.71 (4.21)	2.35 (4.11)	4.12 (0.50)	1.90 (0.08)	7.34 (2.62)
	提案	20.31 (29.12)	**27.85** (3.39)	1.57 (2.40)	4.09 (0.50)	**1.65** (0.15)	**4.01** (2.45)
交互	固定	-18.64 (72.83)	21.75 (7.26)	4.29 (6.28)	5.04 (0.89)	N/A N/A	N/A N/A
	SAC	6.60 (71.61)	15.82 (9.42)	2.72 (5.39)	4.65 (1.33)	1.90 (0.08)	7.38 (2.64)
	提案	**19.53** (51.30)	**14.90** (8.22)	**1.75** (3.28)	**4.37** (1.15)	**1.71** (0.17)	**4.43** (2.51)

$B = 256$ 的小批次。针对每个小批次计算所有损失函数，并使用其梯度，通过采用默认配置的 AdaTerm [38] 优化下文定义的所有网络参数。

关于待优化函数的设计，两个价值函数 $Q_{1,2}$ 由独立的神经网络近似，每个网络包含两个隐藏层，每层有 100^{-} 个神经元。它们的目标网络 $Q_{1,2}$采用默认配置进行 CAT 软更新 [25]。策略 π 由对角多元 Student-t 分布建模，其参数（即位置参数、尺度参数和自由度）由共享神经网络近似，输出层除外；这些网络的架构与 $Q_{1,2}$相同。松弛变量 Δ 也由神经网络近似，其架构与 $Q_{1,2}$ 和π相同。采用文献 [29], [30] 提出的压缩函数作为这些网络的激活函数。此外，为实现网络归一化，在激活函数之前应用 RMSNorm [39]（原为 LayerNorm）。

请注意，SAC 会将由 π 生成的动作（其基本上位于实数空间中）转换为盒约束动作 $a \in [-1, 1]^{|\mathcal{A}|}$。通常使用双曲正切函数进行这种转换，但它往往会使动作集中在边界附近，并产生较大的梯度。为获得更稳定的梯度，我们采用了基于文献 [29], [30], 的另一种转换方法，该方法也用于 Δ。

作为超参数之一，α 的初始值设为 1。α （更具体地说，其实数版本 $\tilde{\alpha}$）与网络一起进行优化。式（13）中的 ϵ简单地调节为 $\epsilon = 0.1|\mathcal{A}|$，因为关于 Δ 的等式约束中的所有项都与 $|\mathcal{A}|$成正比。

B. 实验配置

对于可变阻抗任务，使用了图7所示的五自由度机械臂。它由 HEBI Robotics 开发的执行器（及机架）组成，每个执行器都是串联弹性执行器（SEA）。因此，可以根据内部弹簧两侧旋转角度的差值计算所施加的扭矩，并据此进行扭矩控制。

在此之后的实验中，仅使用了 HEBI Robotics 提供的官方控制器：即通过控制策略4控制目标位置和速度；进行重力补偿；施加偏置，以大致补偿安装在肩部的气弹簧；以及为每个轴配备虚拟弹簧和阻尼器的任务空间阻抗控制器。PD增益设得低于默认值，因此会有意产生跟踪误差，而阻抗控制器的补偿则有助于提升性能。具体而言，腕部的两个执行器的比例增益设为5，其余三个执行器的比例增益设为20，所有执行器的微分增益均设为0.01。

对于该机械臂，末端执行器的目标轨迹设为以下正弦波。

$$p^{\mathrm{tar}}(t) = p^{\mathrm{ini}} \pm A \sin(\omega \pi t) \tag{15}$$

其中，t 表示从回合开始起经过的时间（控制周期为 20 ms），p^{ini} 由各关节范围中心位置的正向运动学计算得出，作为末端执行器的初始位置。A 表示各轴的振幅：$A_{xy} \in [0.05, 0.15]^2$ 对应水平面，$A_z \in [0.01, 0.05]$ 对应垂直方向。所有轴的 $\omega \in [0.1, 0.9]^3$ 均设为轴向相关频率。每个回合中，A、ω和符号 $\pm$ 均独立地进行均匀随机化。为跟踪 p^{tar}，使用逆运动学计算目标关节角度 q^{tar} 和速度$\dot{q}^{\mathrm{tar}}$，并将 [40], 发送至各执行器作为控制指令。

在上述用于跟踪给定轨迹的阻抗控制系统下，强化学习（即 SAC 和所提方法）对阻抗参数进行优化（即三维虚拟弹簧 $k_p \in [0, 100]^3$ 和阻尼器 $k_d \in [0, 10]^3$）。为此，动作空间被设计为这些参数的变化量（共六个维度），其范围限制在最大值的 ± 5 % 以内。以下奖励函数旨在使跟踪目标轨迹的误差保持在允许范围内，同时尽量减少所需努力。

$$r = \exp\left(-\sum_i |\tau_i|\right) - \begin{cases} 1 & \|p^{\mathrm{tar}} - p^{\mathrm{obs}}\|_2 > 0.05 \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

其中， τ 表示所需努力， p^{obs} 根据观测到的角度通过正向运动学计算得出，即观测到的末端执行器位置。为使这些配置满足马尔可夫决策过程，状态空间共包含 32 个维度，包括每个执行器的角度、角速度、所需努力和温度；末端执行器的观测位置和目标位置；以及当前阻抗参数。

致谢

本研究得到日本JST PRESTO项目的资助，资助编号为JPMJPR20C3。

参考文献

[1] Y. Tsurumine、Y. Cui、E. Uchibe 和 T. Matsubara，“采用平滑策略更新的深度强化学习：在机器人布料操作中的应用”，*Robotics andAutonomous Systems*，第112卷，83，2019年72页。– 。[2] T. Kobayashi、E. Dean-Leon、J. R. Guadarrama-Olvera、F. Bergner 和 G. Cheng，“基于主从切换的全身多接触触觉人-人形机器人交互：机器人的‘盒式步’舞蹈”，*Advanced Intelligent Systems*，第2100038页，2021年。 [3] R. S. Sutton 和 A. G. Barto，强化学习：导论。MIT出版社，2018年。 [4] O. M. Andrychowicz、B. Baker、M. Chociej、R. Jozefowicz、B. McGrew、J. Pachocki、A. Petron、M. Plappert、G. Powell、A. Ray 等，“学习灵巧手内操作”，*The International Journal of Robotics Research*，第39卷，第1期，第3–20页，2020年。 [5] B. Osi´nski、A. Jakubowski、P. Zi. ecina、P. Miło´s、C. Galias、S. Homo- ceanu 和 H. Michalewski，“面向现实世界自动驾驶的基于仿真的强化学习”，载于*IEEE*国际机器人与自动化会议。IEEE，2020年，第6411–6418页。 [6] Z. Li、X. Cheng、X. B. Peng、P. Abbeel、S. Levine、G. Berseth 和 K. Sreenath，“面向双足机器人的鲁棒参数化运动控制的强化学习”，载于*IEEE*国际机器人与自动化会议。IEEE，2021年，第2811–2817页。 [7] T. Haarnoja、A. Zhou、P. Abbeel 和 S. Levine，“软演员-评论家：具有随机演员的离策略最大熵深度强化学习”，载于国际机器学习会议。机器学习研究会论文集，2018年，第1861–1870页。 [8] T. Haarnoja 、 A. Zhou 、 K. Hartikainen 、 G. Tucker 、 S. Ha 、 J. Tan、V. Kumar、H. Zhu、A. Gupta、P. Abbeel 和 S. Levine，“软演员-评论家算法及其应用”，*arXiv*预印本 *arXiv:1812.05905*，2018年。 [9] B. Eysenbach 和 S. Levine，“最大熵强化学习（可证明地）解决某些鲁棒强化学习问题”，载于国际学习表征会议，2021年。 [10] P. Christodoulou，“离散动作设置下的软演员-评论家”，*arXiv*预印本 *arXiv:1910.07207*，2019年。 [11] P. N. Ward 、 A. Smofsky 和 A. J. Bose，“ 利用归一化流策略改进软演员-评论家中的探索”，*arXiv*预印本 *arXiv:1906.02771*，2019年。 [12] Q. Yang、T. D. Simão、S. H. Tindemans 和 M. T. Spaan，“WCSAC：面向安全约束强化学习的最坏情况软演员-评论家”，载于*AAAI*人工智能会议论文集，第35卷，第12期，2021年，第10 639–10 646页。 [13] S. Levine，“作为概率推断的强化学习与控制：教程与综述”，*arXiv*预印本 *arXiv:1805.00909*，2018年。 [14] E. Liang、R. Liaw、R. Nishihara、P. Moritz、R. Fox、K. Goldberg、J. Gonzalez、M. Jordan 和 I. Stoica，“RLLib：分布式强化学习的抽象”，载于国际机器学习会议。机器学习研究会论文集，2018年，第3053–3062页。 [15] A. Raffin、A. Hill、A. Gleave、A. Kanervisto、M. Ernestus 和 N. Dor- mann，“Stable-Baselines3：可靠的强化学习实现”，机器学习研究期刊，第22卷，8，2021年268，1第。页– 。[16] Y. Wang 和 T. Ni，“Meta-SAC：通过元梯度自动调节软演员-评论家的熵温度”，*arXiv* 预印本 *arXiv:2007.01932*，2020年。 [17] J. Lin、C. Huang、X. Liang 和 L. Lin，“Cat-SAC：具有好奇心感知熵温度的软演员-评论家”，2020年。 [18] Y. Xu、D. Hu、L. Liang、S. McAleer、P. Abbeel 和 R. Fox，“离散软演员-评论家的目标熵退火”，*arXiv*预印本 *arXiv:2112.02852*，2021年。 [19] T. P. Lillicrap、J. J. Hunt、A. Pritzel、N. Heess、T. Erez、Y. Tassa、D. Silver 和 D. Wierstra，“利用深度强化学习进行连续控制”，载于国际学习表征会议，2016年。 [20] S. Fujimoto、H. Hoof 和 D. Meger，“解决演员-评论家方法中的函数逼近误差”，载于国际机器学习会议。机器学习研究会论文集，2018年，第1587–1596页。 [21] S. Boyd、S. P. Boyd 和 L. Vandenberghe，《凸优化》。剑桥大学出版社，2004年。 [22] V. Vapnik，《统计学习理论的本质》。施普林格科学与商业媒体，1999年。 [23] E. Todorov、T. Erez 和 Y. Tassa，“Mujoco：用于基于模型控制的物理引擎”，载于IEEE/RSJ智能机器人与系统国际会议。IEEE，2012年，第5026–5033页。 [24] E. Coumans 和 Y. Bai，“ PyBullet：用于游戏、机器人学和机器学习物理仿真的Python模块”，GitHub代码库，2016年。[25] T. Kobayashi，“面向深度强化学习的整合自适应t-soft更新”，arXiv预印本 arXiv:2202.12504，2022年。 [26] D. P. Kingma 和 M. Welling，“自编码变分贝叶斯”，载于国际学习表征会议，2014年。 [27] A. Beck 和 M. Teboulle，“凸优化的镜像下降与非线性投影次梯度方法”，《运筹学快报》，第31卷，第3期，第167–175页，2003年。 [28] V. Picheny、R. B. Gramacy、S. Wild 和 S. Le Digabel，“带松弛变量增广拉格朗日的混合约束下贝叶斯优化”，《神经信息处理系统进展》，第29卷，2016年。[29] J. T. Barron，“ Squareplus：类Softplus代数整流函数”，arXiv预印本 arXiv:2112.11687，2021年。 [30] T. Kobayashi 和 T. Aotani，“面向任意随机策略且具有计算效率的受限归一化流设计”，《Advanced Robotics》，第1–18页，2023年。 [31] J. C. Bezdek 和 R. J. Hathaway，“交替优化的收敛性”，《神经、并行与科学计算》，第11卷，第4期，第351–368页，2003年。 [32] C. Tessler、Y. Efroni 和 S. Mannor，“连续控制中的动作鲁棒强化学习及其应用”，载于国际机器学习会议。机器学习研究会论文集，2019年，第6215–6224页。 [33] A. Ben-Tal、L. El Ghaoui 和 A. Nemirovski，《鲁棒优化》。普林斯顿大学出版社，2009年，第28卷。 [34] T. Kobayashi，“L2C2：面向稳定平滑强化学习的局部利普希茨连续约束”，载于IEEE/RSJ智能机器人与系统国际会议，2022年，第4032–4039页。 [35] J. Buchli、F. Stulp、E. Theodorou 和 S. Schaal，“学习可变阻抗控制”，The International Journal of Robotics Research，第30卷，第7期，第820–833页，2011年。 [36] F. Ficuciello、L. Villani 和 B. Siciliano，“面向直观人机物理交互的冗余机械臂可变阻抗控制”，《IEEE机器人汇刊》，第31卷，第4期，第850–863页，2015年。 [37] A. Paszke、S. Gross、S. Chintala、G. Chanan、E. Yang、Z. DeVito、Z. Lin、A. Desmaison、L. Antiga 和 A. Lerer，“PyTorch中的自动微分”，载于神经信息处理系统进展研讨会，2017年。 [38] W. E. L. Ilboudo、T. Kobayashi 和 M. Takamitsu，“AdaTerm：面向抗噪随机梯度优化器的自适应t分布估计鲁棒矩”，arXiv预印本 arXiv:2201.06714，2022年。 [39] B. Zhang 和 R. Sennrich，“均方根层归一化”，神经信息处理系统进展，第32卷，2019年。 [40] T. Kobayashi，“面向盒约束关节空间的镜像下降逆运动学”，arXiv预印本 arXiv:2101.07625，2021年。