2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Multi-Alpha Soft Actor-Critic: Overcoming Stochastic Biases in Maximum Entropy Reinforcement Learning

Conor Igoe, Swapnil Pande, Siddarth Venkatraman, Jeff Schneider

Abstract— **The successful application of robotic control requires intelligent decision-making to handle the long tail of complex scenarios that arise in real-world environments. Recently, Deep Reinforcement Learning (DRL) has provided a data-driven framework to automatically learn effective policies in such complex settings. Since its introduction in 2018, Soft Actor-Critic (SAC) remains as one of the most popular off-policy DRL algorithms and has been used extensively to learn performant robotic control policies. However, in this paper we argue that by relying on the maximum entropy formalism to define learning objectives, previous work introduces a significant bias away from optimal decision making, which often requires near-deterministic behaviour for high-precision tasks. Moreover, we show that when training with the original variants of SAC, overcoming this bias by reducing entropy budgets or entropy coefficients introduces separate issues that lead to slow or unstable learning. We address these shortcomings by treating the entropy coefficient α as a random variable and introduce Multi-Alpha Soft Actor-Critic (MAS). We show how MAS overcomes the stochastic bias of SAC in a variety of robotic control tasks including the CARLA urban-driving simulator, while maintaining the stability and sample efficiency of the original algorithms.**

I. Introduction

In recent years, there has a been an explosive research effort towards the goal of designing Deep Reinforcement Learning algorithms (DRL) that automatically learn effective robotic control policies [1]. However, the real world is complex. Consequently DRL typically requires a large number of environment interactions to learn competitive behaviours. From surgical robotics to self-driving vehicles, environment interactions are costly, leading to a growing interest in building sample efficient learning algorithms.

Crudely, the space of DRL algorithms can be divided into on- and off-policy methods [2], with off-policy algorithms typically requiring considerably less environment interactions to learn performant policies compared to on-policy methods. Soft Actor-Critic (SAC) [3] has emerged as one of the most popular off-policy DRL algorithms and has been used by several authors to achieve state-of-the-art policies in robotic control benchmark tasks [3], [4].

Standard reinforcement learning algorithms focus on optimising policies with respect to long-term expected rewards $\mathbb{E}[\sum_t r_t]$. However, taking inspiration from the Maximum Entropy formalism (Max-Ent) [5], SAC explicitly incentivises stochastic decision making by augmenting rewards with policy entropies, $\mathbb{E}[\sum_t r_t + \alpha\mathscr{H}(\pi(\cdot|s_t))]$. The Max-Ent formalism has been extensively studied from both theoretical [6] and applied perspectives [7]–[10], and has been found to have three particularly attractive features. Specifically, incentivising long-term stochastic decision-making during learning:

- results in a sequence of policies that functions as a convenient exploration mechanism;
- improves training stability over similar algorithms that do not include long-term entropy components in their training objectives;
- and yields policies that exhibit robustness to test-time environment distributional shifts.

Despite these important advantages over the standard reinforcement learning formalism, in this paper, we highlight an issue that arises when training policies using SAC for high-precision control tasks. In particular, explicitly incentivising stochastic policies via soft objective functions biases agents away from optimal behaviour that requires (near-)deterministic decision making. We refer to this bias as the Stochastic Bias. Figure 1 illustrates the effect of this bias for a simple 2D Toy Hop environment where the goal is to hop as far along the x-axis as possible by taking hops within a radius of $\delta_{\max}$. Intuitively, increasing the entropy requirement limits how large a hop the agent can risk taking, resulting in the agent taking safer but shorter hops. Further details of this Markov Decision Process (MDP) are provided in Section IV-A.

In addition, policies trained with SAC are typically evaluated using a deterministic conversion [3], [11]. This is usually achieved by acting according to the policy's mean action in a given state. However, this conversion introduces a distribution shift between training and testing and can result in evaluation policies that act in vain anticipation of future stochasticity. The Stochastic Bias is especially problematic for self-driving applications, where, for example, turning sharp corners and navigating narrow streets at high speeds can incur significant costs when acting stochastically. Consequently, using SAC to train self-driving agents can induce excessively cautious and slow driving styles [4]. Moreover, although naively reducing the entropy requirement in SAC (either by changing α or by changing the target entropy) reduces this bias, it comes at the costs of reduced exploration, increased training instability, and reduced robustness to environment shifts [12], [13].

C. Igoe is with the Machine Learning Department, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213; S. Pande, S. Venkatraman and J. Schneider are with the Robotics Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 {cigoe, siddartv, spande, schneide}@cs.cmu.edu

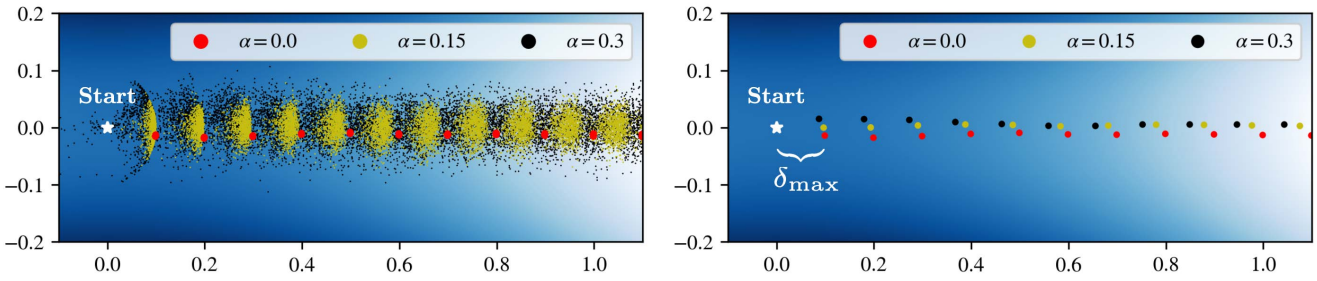

Fig. 1: **Illustration of the Stochastic Bias** The figure on the left (right) shows the state visitation distributions of π_*^α (`MakeDeterministic`(π_*^α)) in the ToyHop MDP for 3 different α's trained using MAS. Note that increasing α has two effects on the visitation distribution: the variance increases, and the mean hop length decreases from the maximum allowable hop length $\delta_{\max}$. Note also that the blue gradient in the figure background shows the reward function $r(s) = s_x^2 - s_y^2$, with darker shades corresponding to lower rewards. We remark that the clusters of hop locations can be plotted arbitrarily close as a function of α but for $\alpha > 0.3$, overlapping clusters hides the pattern we are interested in highlighting.

The central argument of this paper is that, when training using SAC, there may not exist a fixed α for which we can reap the myriad of benefits of the Max-Ent setting while simultaneously avoiding the shortcomings associated with stochastic policies. Leveraging this insight, we introduce Multi-Alpha Soft-Actor Critic (MAS), an off-policy DRL algorithm that treats α as a random variable. By using randomly generated α's in both exploration and training, we avoid the need to commit to or converge to a single α during learning. We show through experiments on the CARLA [14] urban driving simulator how policies learned using MAS achieve higher mean-driving speeds than SAC policies with the same amount of environmental experience. In addition, we describe an illustrative MDP that highlights the Stochastic Bias induced by the soft setting, in which MAS performs exceptionally well compared to SAC. Finally, we provide experimental results on two popular DRL benchmarks, including a robotic manipulation task from the Panda-Gym library [15], where the the effects of the Stochastic Bias are clear.

II. Related Work

Interest in the Maximum Entropy framework (Max-Ent) for decision making under uncertainty stems back to work in Inverse Reinforcement Learning [5]. Subsequent papers provided empirical evidence supporting the claims that Max-Ent RL offered valuable robustness properties over standard RL approaches ([7]–[10], [12]) and recent work provided theoretical insights into the nature of this robustness [6]. Several authors have proposed both on- and off-policy model-free DRL algorithms that draw direct inspiration from the Max-Ent formalism [13], [16]–[18]. Soft Actor-Critic [11] emerged as a popular off-policy Max-Ent DRL algorithm due in part to its strong sample complexity without requiring the use of complex approximate inference procedures in continuous action spaces. We refer to this version of SAC as SACV0. As described in the original paper, SACV0's performance was found to be strongly dependent on the entropy coefficient α. A follow-up paper [3] was published by the same group describing a variant of SAC that automatically tuned α by considering a constrained version of the Max-Ent objective and approximating the solution to the associated dual optimisation problem. We refer to this follow-up version of SAC as SACV1. The authors in the follow-up paper additionally proposed a heuristic to define entropy targets, although it is common to additionally tune this entropy target in practice. As described in multiple papers [19]–[21], the performance of DRL algorithms is highly sensitive to choices of hyperparameters, such as α in SACV0 and $\bar{\mathcal{H}}$ in SACV1. Several papers propose meta-RL algorithms [22]–[24] that effectively use experience in a family of MDPs to learn how to automatically change hyperparameters to adapt to a new MDP. Although work in meta-RL typically considers a distribution over the training environments, the approach taken in this paper differs in that we do not use observations from the evaluation rollout to adapt behaviour to a given test scenario. To the best of the authors' knowledge, the only work that explicitly treats RL hyperparameters as a random variable during training is Agent57 [25]. However, the authors only consider a distribution over γ, which has an entirely different motivation than what we consider here.

A motivating example of a precision control task that has recently garnered significant attention is the problem of RL for self-driving cars. Several authors have demonstrated strong results on autonomous driving car benchmarks via the use of model-free RL algorithms [4], [26]–[28], with various parameterisations of the MDP and choice of training algorithms. In particular, [4] proposes a parallelised implementation of SACV0, which achieves performance on par with the state-of-the-art. However, the work also demonstrates that performance of the policy is highly sensitive to choice of α, with values of α that are too large or too small inducing policies that drive slowly and completely stop driving at tight bends. Examples like this illustrate the practical utility of algorithms that can mitigate the negative effects of the Stochastic Bias.

III. Multi-Alpha Soft Actor-Critic

We first introduce notation and summarize the standard and maximum entropy reinforcement learning frameworks.

A. Notation & Preliminaries

We address policy learning in continuous action spaces. We consider an infinite-horizon Markov Decision Process, defined by the tuple $\mathscr{M} = (\mathscr{S}, \mathscr{A}, p, r, \mu)$, where the state space $\mathscr{S}$ and the action space $\mathscr{A}$ are continuous. The initial state $\mathbf{s}_0 \in \mathscr{S}$ is assumed to be drawn independently from probability density $\mu : \mathscr{S} \to [0, \infty)$ at the start of each environment rollout, and the state transition probability $p : \mathscr{S} \times \mathscr{S} \times \mathscr{A} \to [0, \infty)$ represents the probability density of the next state $\mathbf{s}_{t+1} \in \mathscr{S}$ given the current state $\mathbf{s}_t \in \mathscr{S}$ and action $\mathbf{a}_t \in \mathscr{A}$. The environment emits a bounded reward $r : \mathscr{S} \times \mathscr{A} \to [r_{\min}, r_{\max}]$ on each transition. We will use $\rho_\pi(\mathbf{s}_t)$ and $\rho_\pi(\mathbf{s}_t, \mathbf{a}_t)$ to denote the state and state-action visitation distributions induced by a policy $\pi(\mathbf{a}_t|\mathbf{s}_t)$.

Standard RL [2] maximizes the expected sum of rewards $\mathbb{E}_{\mathbf{s}_t, \mathbf{a}_t \sim \rho_\pi}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right]$. The Maximum Entropy formalism [11] augments this objective by introducing policy entropy components weighted by a coefficient α:

$$J(\pi; \alpha) = \mathbb{E}_{\mathbf{s}_t, \mathbf{a}_t \sim \rho_\pi}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathscr{H}\big(\pi(\cdot|\mathbf{s}_t)\big)\right] \tag{1}$$

where $\mathscr{H}(\pi(\cdot|\mathbf{s}_t))$ denotes the differential entropy of the probability distribution over actions defined by policy π at state $\mathbf{s}_t$. For ease of exposition, we also use the shorthand $\mathscr{H}_t = \mathscr{H}(\pi(\cdot|\mathbf{s}_t))$ and $r_t = r(\mathbf{s}_t, \mathbf{a}_t)$. Both settings also typically make use of a scalar discount factor $\gamma \in [0, 1]$ to control the effective lookahead horizon of the agent, as in $\sum_t \gamma^t r_t$ and $\sum_t \gamma^t (r_t + \alpha \mathscr{H}_t)$. We use π^α_* to denote a policy that maximises (1) for a given α.

Amongst other contributions, the original Soft Actor-Critic paper [11] proposes a family of loss functions for actor and critic networks π_ϕ and Q_θ inspired by the objective in (1). In the interest of space, we restate the the SACV0 loss functions here and refer the interested reader to the original paper for details on their derivation. The (regular) critic's loss function is given by:

$$J_Q(\theta) = \tag{2}$$
$$\mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1}) \sim \mathscr{D}}\left[\frac{1}{2}\left(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \Big(r_t + \right.\right.$$
$$\left.\left.\gamma \mathbb{E}_{\mathbf{a}_{t+1}}\left[Q_{\bar{\theta}}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) - \alpha \log \pi(\mathbf{a}_{t+1}|\mathbf{s}_{t+1})\right]\Big)\right)^2\right]$$

where $\mathscr{D}$ is a replay buffer of exploration experience, $\mathbf{a}_{t+1}$ is resampled from the policy $\pi_\phi(\cdot|\mathbf{s}_{t+1})$ and $Q_{\bar{\theta}}$ denotes the target critic that uses parameters $\bar{\theta}$ which are slowly updated to track the regular critic's parameters to stabilise learning. Additionally, to reduce maximisation bias, SACV0 also makes use of two regular critics and two target critics. In particular, the minimum of both target critics are used in place of the single target critic in (2), and each regular critic is trained using (2) from independent random initialisations. The policy's loss function is given by:

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathscr{D}}\left[\mathbb{E}_{\mathbf{a}_t \sim \pi_\phi(\cdot|\mathbf{s}_t)}\left[\alpha \log \pi_\phi(\mathbf{a}_t|\mathbf{s}_t) - Q_\theta(\mathbf{s}_t, \mathbf{a}_t)\right]\right] \tag{3}$$

where, as in (2), the minimum of the two regular critic is used in place of the single regular critic. Typically, the policy is parameterised as a (transformed) multivariate diagonal Gaussian, and the reparameterisation trick is used to estimate the gradient of (3). The SACV0 algorithm alternates between collecting experience from the environment with the current policy and updating the actor and critic networks using stochastic gradients from minibatches sampled from $\mathscr{D}$.

An important detail of SACV0 that is of relevance to our work is the assumption of a fixed α throughout training for a given random seed. As described in the original paper, performance is highly sensitive to the choice of α. In an effort to address the problem of choosing an appropriate α in the Max-Ent setting, the authors proposed a related problem formulation in a follow-up paper [3]:

$$\max_\pi J'(\pi) = \max_\pi \mathbb{E}_{\mathbf{s}_t, \mathbf{a}_t \sim \rho_\pi}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right] \tag{4}$$
$$\text{s.t. } \mathbb{E}_{\mathbf{s}_t \sim \rho_\pi}\left[\mathscr{H}\big(\pi(\cdot|\mathbf{s}_t)\big)\right] \geq \bar{\mathscr{H}}. \tag{5}$$

The optimisation problem in (4) led to the "auto-tuning" variant of SAC, SACV1. Similar to SACV0's dependency on α, performance with SACV1 is highly dependent on an appropriate choice of entropy target, $\bar{\mathscr{H}}$. The authors proposed the heuristic $\bar{\mathscr{H}} = -\dim(\mathscr{A})$ in the SACV1 paper, although in practice it is common to tune $\bar{\mathscr{H}}$. Lastly, we refer to $\mathbb{E}\left[\sum_t \gamma^t r_t \middle| \pi\right]$ as the regular value of policy π, $\mathbb{E}\left[\sum_t \gamma^t (r_t + \alpha \mathscr{H}_t) \middle| \pi\right]$ as the soft value of policy π, and $\mathbb{E}\left[\sum_t \gamma^t \mathscr{H}_t \middle| \pi\right]$ as the entropic value of policy π.

B. Multi-Alpha Soft Actor-Critic

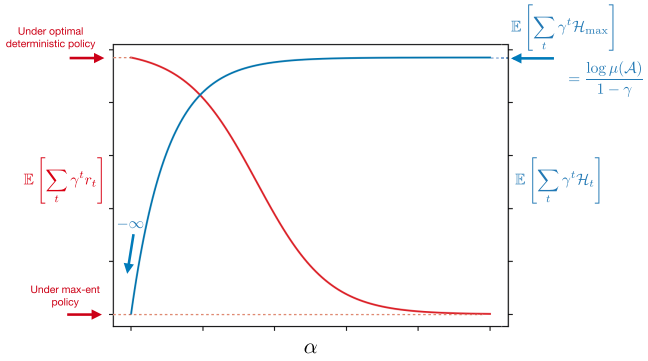

Fig. 2: **Illustration of the Effects of Changing α** Note the two inherently different scales for regular values and entropic values. Note also that the region of steepest change in regular values depends on $\mathscr{M}$. In this illustration we have assumed that all policies in $\arg\max_\pi J(\pi; 0)$ are deterministic for $\mathscr{M}$, and that $\mathscr{A}$ is compact.

As illustrated in Figure 2, when training using SACV0 within MDP $\mathscr{M}$, α plays an important role in controlling the strength of the stochastic bias on soft-optimal policy $\pi^\alpha_* = \arg\max_\pi J(\pi; \alpha)$. The red curve illustrates how, as α increases from 0, the regular value of π^α_* decreases from the maximum achievable regular value in $\mathscr{M}$, asymptotically approaching the regular value associated with the maximum entropy policy. Note that the maximum entropy policy is the policy that samples actions according to the maximum

entropy distribution for every $\mathbf{s} \in \mathscr{S}$. When $\mathscr{A}$ is compact, this policy corresponds to Uniform$(\mathscr{A})\,\forall \mathbf{s} \in \mathscr{S}$. Meanwhile, the blue curve illustrates how the opposite relation holds for entropic values: as $\alpha \to 0^+$, the entropic value for policy π_*^α approaches its infimum, which may be $-\infty$ if all optimal policies in $\arg\max_\pi J(\pi;0)$ are deterministic. As $\alpha \to \infty$, the entropic values converge to the maximum cumulative discounted entropies, given by $\frac{\log \mu(\mathscr{A})}{1-\gamma}$ (assuming $\mathscr{A}$ is compact).

Although the exact shape and range of each curve depends intimately on $\mathscr{M}$, Figure 2 illustrates the inevitable trade-off that occurs when committing to a single value of α, and motivates the use of a distribution over α throughout training. By considering a family of α's, our agent can learn a family of policies, generating exploratory behaviour for large α, but simultaneously learning to extract performant near-deterministic policies for small α. We thus benefit from the Max-Ent setting while avoiding the explicit stochastic bias that it naively entails.

Specifically, we propose 4 additional steps to SACV0:

1) Define a distribution $\mathbb{P}$ over α
2) Condition all networks $(\pi_\phi, Q_{\theta_1}, Q_{\theta_2}, Q_{\bar{\theta}_1}, Q_{\bar{\theta}_1})$ on α
3) Sample $\alpha \sim \mathbb{P}$ for every minibatch $\{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1})_i\}_{i=1}^M \sim \mathscr{D}$ when estimating gradients
4) Sample $\alpha \sim \mathbb{P}$ every timestep during exploration

With slight abuse of notation, we modify the SACV0 loss functions to allow for conditioning on different values of α, giving us the MAS loss functions:

$$J_Q(\theta;\alpha) = \tag{6}$$
$$\mathbb{E}\left[\frac{1}{2}\left(Q_\theta(\mathbf{s}_t,\mathbf{a}_t;\alpha) - \left(r_t + \gamma \mathbb{E}_{\mathbf{a}_{t+1}}\left[Q_{\bar{\theta}}(\mathbf{s}_{t+1},\mathbf{a}_{t+1};\alpha) - \alpha \log \pi(\mathbf{a}_{t+1}|\mathbf{s}_{t+1};\alpha)\right]\right)\right)^2\right]$$

$$J_\pi(\phi;\alpha) = \tag{7}$$
$$\mathbb{E}_{\mathbf{s}_t \sim \mathscr{D}}\left[\mathbb{E}_{\mathbf{a}_t \sim \pi_\phi(\cdot|\mathbf{s}_t;\alpha)}\left[\alpha \log \pi_\phi(\mathbf{a}_t|\mathbf{s}_t;\alpha) - Q_\theta(\mathbf{s}_t,\mathbf{a}_t;\alpha)\right]\right]$$

where, as in SACV0, $\mathbf{a}_{t+1}$ is resampled from the policy $\pi_\phi(\cdot|\mathbf{s}_{t+1};\alpha)$. We make use of two regular and target critics, and use the reparameterisation trick to compute policy gradients. For readability, we have suppressed the outer expectation's subscript in (6), although we assume this expectation is across the same random variables as in (2). The complete Multi-Alpha Soft Actor-Critic algorithm is described in Algorithm 1.

Algorithm 1 Multi-Alpha Soft Actor-Critic

Initialise parameter vectors $\phi, \theta_1, \theta_2, \bar{\theta}_1, \bar{\theta}_2$

for each iteration **do**
  for each environment step **do**
    $\alpha \sim \mathbb{P}$
    $\mathbf{a}_t \sim \pi_\phi(\cdot|\mathbf{s}_t;\alpha)$
    $\mathbf{s}_{t+1} \sim p(\cdot|\mathbf{s}_t,\mathbf{a}_t)$
    $r_t = r(\mathbf{s}_t,\mathbf{a}_t)$
    $\mathscr{D} \leftarrow \mathscr{D} \cup \{(\mathbf{s}_t,\mathbf{a}_t,r_t,\mathbf{s}_{t+1})\}$
  for each gradient step **do**
    $\alpha \sim \mathbb{P}$
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i;\alpha)$ for $i \in \{1,2\}$
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi;\alpha)$
    $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$

IV. Experimental Results

In the following experiments, we use 7 variants of MAS. Each variant uses a different definition of $\mathbb{P}$ when sampling α but are otherwise identical. Each variant's $\mathbb{P}$ definition is described in Table I. Note also that, beyond a critical point unique to the given MDP, larger values of α have diminishing effect on the entropic value. In order to avoid generating samples of α that are excessively high, we propose a Projection Method which takes uniform samples of α and "projects" them through an approximation to the red curve in Figure 2, similar to Inverse Transform Sampling. For the "Proj" variants of MAS, we additionally train an α-conditioned network that approximates the expected regular return $\mathbb{E}[\sum_t \gamma^t r_t | \pi_*^\alpha]$. We then use $V_\theta(\alpha)$ to define a distribution $\mathbb{P}$ that concentrates over regions associated with large changes in regular return. Specifically, we define the function $f(\alpha) = \left|\frac{dV_\theta(\alpha)}{d\alpha}\right|$ and interpret f as an unnormalised probability distribution, which we use as our definition of $\mathbb{P}$. Our implementation of this method involves generating a large sequence of M points between 0 and and a practical upper limit c, $\{\tilde{\alpha}_i = i\frac{c}{M}\}_{i=0}^M$. We then approximate $f(\tilde{\alpha}_i)$ with a finite difference, denoted $\hat{f}(\tilde{\alpha}_i)$, and define $\mathbb{P}$ as the discrete probability distribution over the set $\{\tilde{\alpha}_i\}_{i=0}^M$, such that $\mathbb{P}(\alpha = \tilde{\alpha}_i) = \frac{\hat{f}(\tilde{\alpha}_i)}{\sum_j \hat{f}(\tilde{\alpha}_j)}$. We use the notation Proj$([0,c])$ to denote the MAS variant that uses the Projection Method along with an upper bound of c.

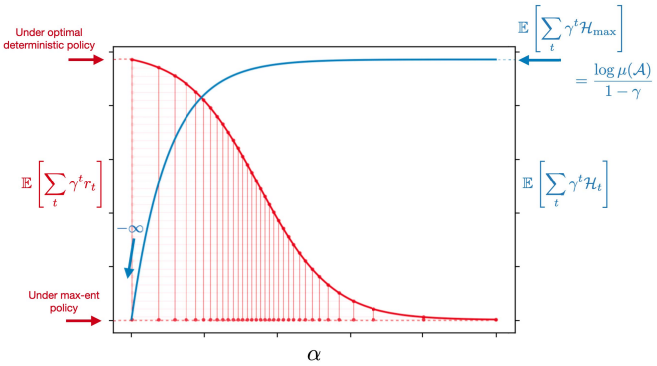

Fig. 3: **Illustration of the Projection Method** Each red point along the horizontal axis is equally spaced out in terms of cumulative probability, when interpretting $f(\alpha) = \left|\frac{d\mathbb{E}[\sum_t \gamma^t r_t | \pi_*^\alpha]}{d\alpha}\right|$ as an unnormalised probability density function. Note that the effect is to concentrate points in regions where there is a significant change in the expected regular return.

For our baselines, we use SACV0 and SACV1 with "high"

"regular" or "low" entropy. For SACV0RegEnt, we tune α using grid search to optimise for mean evaluation return after 5 million timesteps of environment interactions with SACV0 using 5 random seeds per choice of α. The highest performing α is then taken to define SACV0RegEnt. We then use double this value of α to define SACV0HighEnt, and half this value to define SACV0LowEnt. Similarly, for SACV1RegEnt, we use the heuristic proposed by the original authors ($-\dim(\mathscr{A})$) and tune the reward scale. We then define SACV1HighEnt by using the best reward scale found for SACV1RegEnt and adding 2 to the entropy target given by the original heuristic. For SACV1LowEnt we subtract 2 from the heuristic.

TABLE I: Description of MAS variants

	MAS1	**MAS2**	**MAS3**	**MAS4**
$\mathbb{P}$	Unif([0,2])	Unif([0,1])	Proj([0,1])	Proj([0,2])
	MAS5	**MAS6**	**MAS7**	
$\mathbb{P}$	Unif([0,0.5])	Gamma(1, 0.17)	Gamma(1,0.67)	

A. Toy Hop & Panda-Gym

Our first experiments illustrate how SACV0 and SACV1 struggle to learn in a simple toy environment. The Toy Hop MDP describes an environment where an agent's goal is to "hop" as far along the x-axis as possible, achieving positive rewards the further its distance along the axis and penalties for deviating along the y-axis. Hops are restricted to within a radius of $\delta_{\max}$ from the current agent's location. Full details of the MDP are provided below.

- State $s_t = (x_t, y_t) \in \mathbb{R}^2$ describes the agent's x and y coordinates on the 2D plane.
- Action $a_t = (\theta_t, \delta_t)$ specifies the agent's hop angle measured clockwise from the x-axis in radians $\theta_t \in [-\pi, \pi]$, and hop length $\delta_t \in [0, \delta_{\max}]$.
- The agent's location is updated as per its chosen hop $s_{t+1} = s_t + (\delta_t \sin\theta_t, \delta_t \cos\theta_t)$.
- The agent is always initialised at the origin $s_0 = (0,0)$.
- The agent receives reward $r_t = x_t^2 - y_t^2$.

Clearly, the policy that takes action $(0, \delta_{\max}) \forall \mathbf{s} \in \mathscr{S}$ is optimal for this MDP. Indeed, as shown in Figure 6, every variant of MAS rapidly finds this policy and achieves optimal regular values. In comparison, no variant of SACV0 or SACV1 finds this policy, and only one variant (SACV1VeryLowEnt[1]) comes close. Notably, the best performing variant of SAC learns slower and shows more signs of instability compared to all MAS variants. Moreover, all seeds of SACV1VeryLowEnt converge to a policy that leaves a significant performance gap in terms of regular values compared to all MAS variants. Figure 1 illustrates the learned behaviour from MAS for a variety of α's, clearly demonstrating the reduction in hop length that comes with increased stochasticity. An additional phenomenon exhibited by SAC is the initial increase in regular returns before eventually plummeting as the effects of the stochastic bias are realised. In addition, Figure 4 shows similar results in a separate MDP from the Panda-Gym library [15] ("Panda Reach") which simulates a simple robotic manipulation task.

[1]For SACV1VeryLowEnt, we subtract 4 from the original SACV1 target entropy heuristic.

B. CARLA

To demonstrate the performance of MAS for the autonomous driving problem, we train and evaluate policies for the NoCrash [29] benchmark in CARLA 9.10 [14]. This benchmark requires the agent to drive to a goal location from a fixed initial location in a urban setting, with three levels of increasingly busy traffic conditions. We choose to focus on the empty variant, with no other vehicles on the road as this is sufficient to illustrate the effects of the Stochastic Bias when training with SAC. We train our policies in Town01 and evaluate the performance on unseen routes in Town02. We define MDP similar to the one specified in [3], [4]. We assume access to a sequence of target waypoints generated by a global planner that specify the route to follow. The observation space consists of the average heading error to the next 5 target waypoints $\varphi_{\text{near},t}$ and following 5 target waypoints $\varphi_{\text{far},t}$, signed lateral distance from the trajectory d_t, current velocity of the ego-vehicle v_t, and current steer angle of the ego vehicle β_t. The action space consists of the desired steer angle $\beta_{\text{des},t}$ and desired speed at the next step $v_{\text{des},t}$, which is tracked by a PID controller. At each timestep, the policy receives rewards proportional to its current velocity and a small penalty proportional to its deviation from the target trajectory.

The empty variant of the CARLA benchmark requires the car to drive from an initial location to a goal destination while following basic traffic rules with no other vehicles or pedestrians present. Additionally, the benchmark defines 25 training routes in Town01 and an additional 25 evaluation routes in Town02. As defined by the benchmark, we train policies using the training routes and present results with the evaluation routes in Town02.

We define MDP similar to the one specified in [3], [4]. Given the initial position and goal position, we first apply a global planner that generates a sequence of waypoints spaced approximately 2 meters apart. Using these waypoints, the observation space consists of the following components:

- $\varphi_{\text{near},t} \in [-\pi, \pi]$ describes the average heading error to the next 5 waypoints. Specifically, the heading error is defined as the angle between the heading of the vehicle and the vector between the center of mass of the vehicle to the waypoint;
- $\varphi_{\text{far},t} \in [-\pi, \pi]$ describes the average heading error to the waypoints following the next 5 waypoints (waypoints 6-10);
- $v_t \in [0, v_{\max}]$ describes the current velocity;
- $d_t \in \mathbb{R}$ the signed lateral distance to the trajectory specified by the waypoints.

The action space consists of the following:

- $\beta_{\text{des},t} \in [-1, 1]$ describes the desired steer angle of the vehicle;
- $v_{\text{des},t} \in [-1, 1]$ describes the desired speed. -1 maps to 0 km/hr and 1 maps to $v_{\max}$ and the action space is biased

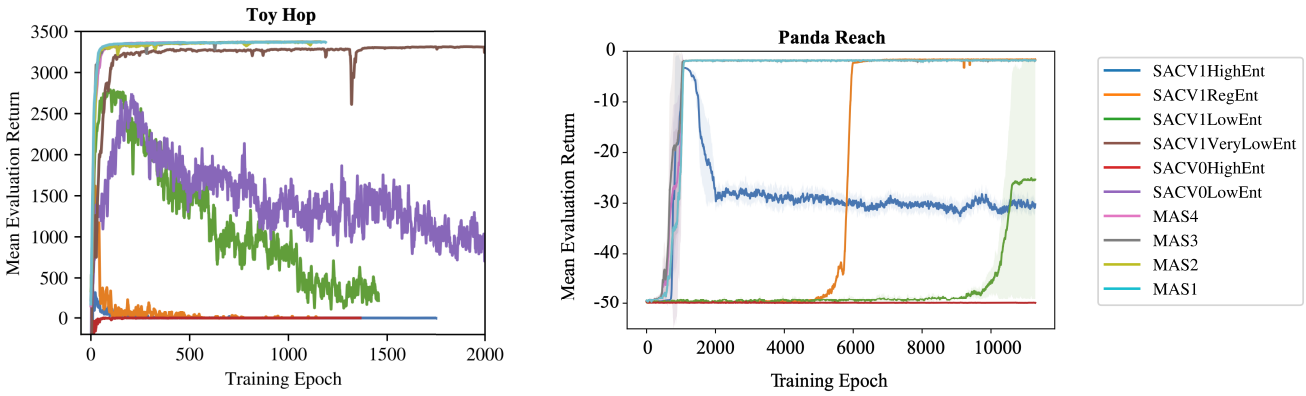

Fig. 4: **Toy Hop & Panda Reach** Each curve is averaged over 10 random seeds, and a moving average filter of size 5 is used for clarity.

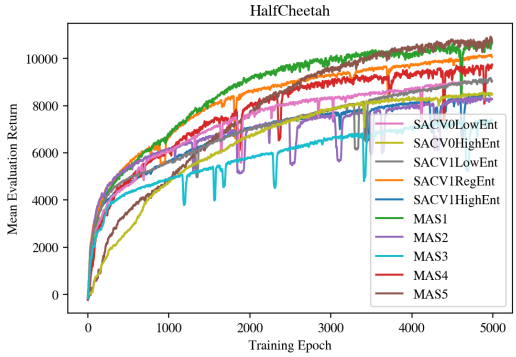

Fig. 5: **Results on HalfCheetah-v2** Each curve is averaged over 5 random seeds.

slightly such that 0 maps to a non-zero velocity. The desired velocity is tracked by a PID controller, which outputs throttle and brake commands.

Finally, the reward function consists of two terms. The agent gets a positive reward at each step, equivalent to the current velocity of the car. The agent also receives a penalty at each step equivalent to the lateral trajectory error d_t. For this problem setting, $v_{\max}$ is defined as 40 km/hr.

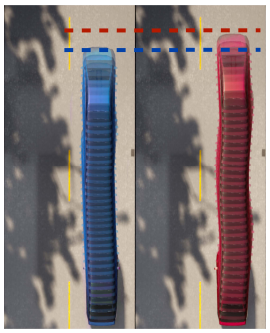

Fig. 6: **Illustration of the Stochastic Bias in CARLA** The blue car is following the best policy trained by all SAC variants in Table II. The red car is following the best policy trained by all MAS variants in Table II. Both policies use the same number of environment interactions.

Table II presents a comparison of the speed achieved

TABLE II: **CARLA Experimental Results**

	SACV1-HighEnt	SACV1-RegEnt	SACV1-LowEnt	MAS1	MAS5	MAS6	MAS7
Average Speed	3.01	3.34	4.7	**6.30**	3.74	2.82	3.19
Std. Dev. Speed	0.85	1.34	1.36	**1.32**	1.56	1.54	1.23

by each policy averaged across 25 pre-determined testing routes in Town02 and 5 random seeds. We find that MAS1 achieves a higher average speed than all SACV1 baselines and other MA variants. Additionally, the standard deviation of the speed for MA1 remains around the same range as all of the other variants, suggesting that the policy did not achieve a higher average simply due to sudden, short bursts of acceleration. Both MAS6 and MAS7, which sample α from a gamma distribution, perform poorly relative to the benchmarks illustrating the importance of tuning the distribution over α. We emphasise that our intention with MAS is not necessarily to eliminate a hyperparameter, but to propose a more effective algorithmic parameterisation compared to variants of SAC. Figure 6 demonstrates a comparison of 3 second trajectories from SACV1-LowEnt and MAS1 from a common intial state in which the vehicle was traveling at a constant velocity. We see that the MAS1 policy traverses a further distance over this 3 second horizon, which compounds into a significant difference over the entire rollout.

C. *HalfCheetah*

We additionally provide experimental results on the popular HalfCheetah benchmark. Figure 5 shows that for a variety of MAS variants, MAS dominates performance of SACV1 and SACV0.

V. Conclusions

In this work, we describe the Stochastic Bias induced by the Maximum Entropy framework. We show how this bias can negatively effect SAC performance for high-precision robotic control tasks, such as in self-driving applications. Moreover, we demonstrate how naive strategies to mitigate these effects introduce additional challenges. We propose Multi-Alpha Soft Actor-Critic to overcome this bias and demonstrate how it effectively learns in MDPs that prove challenging for previous Max-Ent DRL algorithms.

REFERENCES

[1] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine, "How to train your robot with deep reinforcement learning: lessons we have learned," *The International Journal of Robotics Research*, vol. 40, no. 4-5, pp. 698–721, 2021.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[3] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[4] Z. Huang, "Distributed reinforcement learning for autonomous driving," Master's thesis, Carnegie Mellon University, Pittsburgh, PA, May 2022.

[5] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, *et al.*, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[6] B. Eysenbach and S. Levine, "Maximum entropy RL (provably) solves some robust RL problems," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=PtSAD3caaA2

[7] H. J. Kappen, "Path integrals and symmetry breaking for optimal control theory," *Journal of statistical mechanics: theory and experiment*, vol. 2005, no. 11, p. P11011, 2005.

[8] E. Todorov, "Linearly-solvable markov decision problems," *Advances in neural information processing systems*, vol. 19, 2006.

[9] M. Toussaint, "Robot trajectory optimization using approximate inference," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 1049–1056.

[10] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[11] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[12] B. D. Ziebart, *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

[13] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *International conference on machine learning*. PMLR, 2017, pp. 1352–1361.

[14] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 1–16. [Online]. Available: https://proceedings.mlr.press/v78/dosovitskiy17a.html

[15] Q. Gallouédec, N. Cazin, E. Dellandréa, and L. Chen, "panda-gym: Open-Source Goal-Conditioned Environments for Robotic Learning," *4th Robot Learning Workshop: Self-Supervised and Lifelong Learning at NeurIPS*, 2021.

[16] B. O'Donoghue, R. Munos, K. Kavukcuoglu, and V. Mnih, "Combining policy gradient and q-learning," *arXiv preprint arXiv:1611.01626*, 2016.

[17] J. Schulman, X. Chen, and P. Abbeel, "Equivalence between policy gradients and soft q-learning," *arXiv preprint arXiv:1704.06440*, 2017.

[18] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans, "Bridging the gap between value and policy based reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[19] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem, "What matters for on-policy deep actor-critic methods? a large-scale study," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=nIAxjsniDzg

[20] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, F. Janoos, L. Rudolph, and A. Madry, "Implementation matters in deep rl: A case study on ppo and trpo," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=r1etN1rtPB

[21] N. M. Ashraf, R. R. Mostafa, R. H. Sakr, and M. Rashad, "Optimizing hyperparameters of deep reinforcement learning for autonomous driving based on whale optimization algorithm," *Plos one*, vol. 16, no. 6, p. e0252754, 2021.

[22] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, "Rl ̃2: Fast reinforcement learning via slow reinforcement learning," *arXiv preprint arXiv:1611.02779*, 2016.

[23] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick, "Learning to reinforcement learn," *arXiv preprint arXiv:1611.05763*, 2016.

[24] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 1126–1135. [Online]. Available: https://proceedings.mlr.press/v70/finn17a.html

[25] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. D. Guo, and C. Blundell, "Agent57: Outperforming the atari human benchmark," in *International Conference on Machine Learning*. PMLR, 2020, pp. 507–517.

[26] X. Liang, T. Wang, L. Yang, and E. P. Xing, "Cirl: Controllable imitative reinforcement learning for vision-based self-driving," *ArXiv*, vol. abs/1807.03776, 2018.

[27] M. Toromanoff, É. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7151–7160, 2020.

[28] T. Agarwal, H. Arora, and J. G. Schneider, "Affordance-based reinforcement learning for urban driving," *ArXiv*, vol. abs/2101.05970, 2021.

[29] F. Codevilla, E. Santana, A. Lopez, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9328–9337.