Explainable Reinforcement Learning via Dynamic Mixture Policies

Maximilian Schier[1]*, Frederik Schubert[1], and Bodo Rosenhahn[1]

***Abstract*— Learning control policies using deep reinforcement learning has shown great success for a variety of applications, including robotics and automated driving. A key area limiting the adaptation of RL in the real world is the lack of trust in the decision-making process of such policies. Therefore, explainability is a requirement of any RL agent operating in the real world. In this work, we propose a family of control policies that are explainable-by-design regarding individual observation components on object-based scene representations. By estimating diagonal squashed Gaussian and categorical mixture distributions on sub-spaces of the decomposed observations, we develop stochastic policies with easy-to-read explanations of the decision-making process. Our design is generally applicable to any RL algorithm using stochastic policies. We showcase the explainability on an extensive suite of single- and multi-agent simulations, set- and sequence-based high-level scenes, and discrete and continuous action spaces, with performance at least on-par or better compared to standard policy architectures. In additional experiments, we analyze the robustness of our approach to its single additional hyper-parameter and examine its potential for very low computational requirements with tiny policies.**

I. INTRODUCTION

Reinforcement Learning (RL) has shown great performance in robotics [1], automated vehicles [2], [3], [4] and other areas of control, such as drones [5]. While the performance of controllers learned with RL is outstanding, neural network policies trained with RL are inherently difficult to explain, which is an issue for the application of such policies in the real world [6]. Explaining the decision-making process of a controller may likely become an increasingly important legal requirement for its application. In any case, the ability to explain a policy enables the identification of corner cases for failures, and confirm that correct decisions are made for the right reasons. Even if a controller seems to operate correctly in test situations, an explanation may reveal that the behavior was caused through spurious correlations. A well-known example from the computer vision community is many models acting as "Clever Hans"-classifiers on the Pascal VOC data set, identifying horses based on a watermark rather than the rest of the image containing the horse [7]. Similar observations have been made for RL policies operating on images, attending to scores or clocks [8].

While the output of neural network policies can be attributed to individual inputs using saliency maps, this approach can also be problematic [8], and does not reveal how interactions in the input space influence the output space. An explainable RL (XRL) policy should generally be performant, have a direct connection between its explanation and the decision-making process, and offer a good visualization with low cognitive load [9].

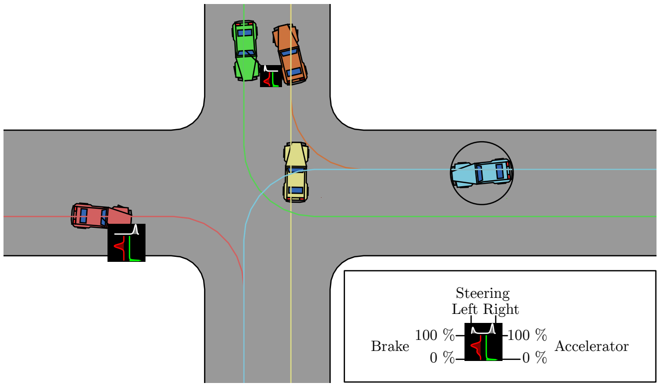

Fig. 1. In this intersection scenario, the blue agent wants to cross the unregulated intersection turning left. Using our proposed mixture distribution policy, its decision-making can be faithfully visualized. Each vehicle is associated with a diagonal squashed Gaussian distribution over the action space to form a stochastic policy, which can be interpreted as shown in the box. The yellow and orange vehicles leaving the intersection are ignored in the decision. For the green vehicle, a slight braking action while steering left is selected, thus the blue vehicle would pass in front of it. However, the red vehicle is given more weight, which is indicated by the increased action explanation visualised as a larger box. Here a stronger braking action while steering right is selected. The resulting action distribution is shown in the box. Braking harder and steering to the right is more likely if the stochastic policy is sampled.

In this work, we propose a novel approach to policies that are intrinsically explainable regarding observed objects, which can be easily adapted to popular RL frameworks using stochastic policies such as SAC [1]. These frameworks commonly use squashed Gaussian or categorical distributions to model the policy on continuous or discrete action spaces, respectively [1], [10]. We propose using mixtures of the corresponding parametric family instead, where each mixture component is estimated using a "myopic" policy observing only a sub-space of the observation space. Thus, this approach effectively implements a post-fusion of the observed sub-spaces, which has several key advantages. With our proposed method the policies are explainable by design and the explanation is faithful, since the resulting action distribution is constructed from the explaining components. In this work, we apply this approach to high-level object perceptions, thus we select each of the dynamic objects as sub-space for one policy. Our main **contributions** are:

- We present a novel formulation of stochastic policies on discrete and continuous action spaces: diagonal

[1] Institute for Information Processing (tnt), L3S, Leibniz University Hannover, Germany.
*Corresponding author, schier@tnt.uni-hannover.de

squashed Gaussian mixture distributions and categorical mixture distributions

- We integrate these distributions into SAC, proposing a novel explainable policy architecture for high-level observations
- We demonstrate that our proposed explainable policies match or outperform the performance of regular policies on common problems and are highly interpretable regarding the influence of individual objects
- Upon acceptance, we publish our implementation

II. RELATED WORK

While mixture distributions have not been used to model an explainable stochastic policy as we propose, they have seen numerous applications in other aspects of reinforcement learning. Mixtures of Gaussians have previously been employed as Q-value- and value-functions. Choi et al. implement a distributional deep Q-network using a parameterized mixture of Gaussians [11], Shahriari et al. follow a similar appraoch for MPO [12], which is closely related to earlier work by Nam et al. for PPO [13]. A Mixture of Gaussians for the critic has also been proposed for Deep Set architectures on dynamically-sized observation spaces [14].

The deconstruction of the action-space based on subspaces of either a statically- or dynamically-sized observation space can also be related to recent advances regarding the improvement of sample-efficiency for Q-functions by using separable Q-functions [15]. Another related field of research is the factorization of joint action spaces in a multi-agent setting with a central critic [16] to learn a robust set of policies for different sub-systems simultaneously.

In the broader topic of explainable RL (XRL) for automotive, Paleja et al. have proposed learning decision trees [17] for continuous action spaces and statically sized observation spaces. Differentiable logic machines on discrete action and observation spaces have been proposed [18]. Generic policies without special design for explainability can to some degree be explained by methods such as saliency maps, which visualize the importance of input regions, most commonly used are pertubation-based saliency [19] or attention-based saliency [20].

Another important current trend in developing explainable policies is using large language models (LLMs) or vision language models (VLMs) to provide natural language explanations for the decision-making of a policy [21]. LLMs and VLMs can achieve good performance for complex data modalities and natural language explanations are easy to understand. However, these models generally require a very large volume of training data, there is a lack of open-source VLMs, the models are not guaranteed to be faithful in reasoning and have very high computational cost for inference [21], [22].

III. APPROACH

In this section, we describe mixture distributions in the Soft Actor-Critic reinforcement learning framework. We then introduce our proposed architecture using parameterized mixture distributions on sub-spaces of the observation for explainable policies.

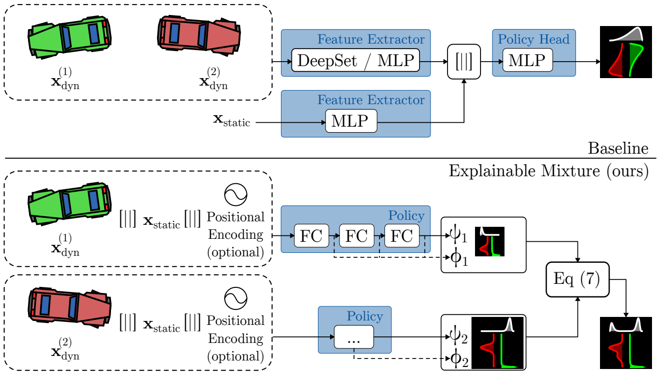
$\mathbf{x}_{\text{dyn}}^{(1)}$
$\mathbf{x}_{\text{dyn}}^{(2)}$
Feature Extractor
DeepSet / MLP
Policy Head
MLP
Feature Extractor
MLP
$\mathbf{x}_{\text{static}}$
Baseline
Explainable Mixture (ours)
$\mathbf{x}_{\text{static}}$
Positional Encoding (optional)
Policy
FC
FC
FC
ψ_1
ϕ_1
Eq (7)
Policy
...
ψ_2
ϕ_2

Fig. 2. Our proposed explainable policies using mixture distributions allow straight-forward interpretation on dynamic scenes. Traditionally, multi-layer perceptrons and Deep Sets are employed for fixed-sized and dynamically-sized observation spaces respectively. We utilize the proposed mixture distribution to independently estimate action distributions per sub space. Positional encodings are employed if the collection is ordered.

A. Soft Actor-Critic

We use the entropy regularized reinforcement learning framework from Soft Actor-Critic (SAC) [1]. The goal of entropy regularized RL is learning a stochastic policy $\pi^* : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ on the observation space $\mathcal{S}$ and action space $\mathcal{A}$, which maximizes the expected entropy regularized return J:

$$\pi^* = \arg\max_\pi J(\pi)\,, \text{ with} \tag{1}$$

$$J(\pi) = \mathop{\mathbb{E}}_{s\sim\mathcal{D},a\sim\pi}\left[Q(s,a) - \alpha \log \pi(a|s)\right], \tag{2}$$

where Q estimates the expected future entropy regularized return following policy π. SAC learns off-policy using samples from the replay buffer $\mathcal{D}$. The entropy regularization coefficient α balances entropy and future-reward. For an in-depth explanation of SAC we refer to [1], [23].

B. Mixture Distributions

Let $p_1(x), \dots, p_k(x)$ be a set of probability density functions. Following notation from [24], for given weights $\phi_1, \dots, \phi_k$ such that $\sum_i \phi_i = 1$, $f(x)$ is a mixture distribution with density $f(x) = \sum_i \phi_i p_i(x)$. If the mixture components are functions of a parametric family, $p(x;\psi)$ parameterized by ψ, we can write $f(x;\psi_1,\dots,\psi_k) = \sum_i \phi_i p(x;\psi_i)$. Furthermore, let $\boldsymbol{\phi} = \left[\phi_1,\dots,\phi_k\right]$, f can be further parameterized with the component weights: $f(x;\boldsymbol{\phi},\psi_1,\dots,\psi_k)$.

C. Mixture Policies on Discrete Action Spaces

On discrete action spaces categorical distributions are commonly used for stochastic policies [25], [10]. Let n be the number of possible actions, and $s \in \mathcal{S}$ be an observation with k observation components. We propose using a parameterized categorical mixture distribution as the policy with the following probability function:

$$b(a;\boldsymbol{\phi},\boldsymbol{\theta}) = \sum_{i=1}^{k} \phi_i \theta_{i,a}\,, \tag{3}$$

where $\boldsymbol{\phi} = \left[\phi_1, \dots, \phi_k\right]$ are the component weights per observation component and $\boldsymbol{\theta} = \left[\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_k\right]$ are the parameters of the mixture distributions, with $\boldsymbol{\theta}_i = \left[\theta_{i,1}, \dots, \theta_{i,n}\right]$ such that $\sum_{j=1}^{n} \theta_{i,j} = 1$. Such a mixture distribution can be sampled from by ancestral sampling:

$$\begin{aligned} z &\sim \mathrm{Categorical}(\cdot; \boldsymbol{\phi}) \\ a &\sim \mathrm{Categorical}(\cdot; \boldsymbol{\theta}_z)\,. \end{aligned}$$

However, this sampling procedure is not differentiable regarding $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ to optimize π using gradient ascent on (2). Instead, we rewrite the objective (2) with the expected value over actions in closed form as in [10]:

$$J(\pi) = \mathop{\mathbb{E}}_{s \sim \mathcal{D}} \left[\sum_{a=1}^{n} \pi(a|s)(Q(s,a) - \alpha \log \pi(a|s)) \right] . \tag{4}$$

D. Mixture Policies on Bounded Continuous Action Spaces

Gaussian distributions are commonly used for continuous action spaces [26], [1], for given mean μ and standard deviation σ the probability density function is:

$$g(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) . \tag{5}$$

Since the Gaussian distribution has infinite support and action spaces are usually bounded, algorithms like SAC apply a squashing [1]. For an unbounded action $u \sim g(\cdot; \mu, \sigma)$ the bounded action is $a = \tanh(u)$. Thus the probability density function with finite support [-1, 1] is:

$$h(a; \mu, \sigma) = \frac{g(\tanh^{-1}(a); \mu, \sigma)}{1 - a^2} , \tag{6}$$

see [1] for a proof. Given an observation with k components and a bounded action $\mathbf{a} \in [-1, 1]^n$. We propose using a diagonal squashed Gaussian mixture distribution for the policy instead, with parameters $\boldsymbol{\phi} \in \mathbb{R}^k$, $\boldsymbol{\mu} \in \mathbb{R}^{k \times n}$, and $\boldsymbol{\sigma} \in \mathbb{R}^{k \times n}$ and the following probability density function:

$$c(\mathbf{a}; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{i=1}^{k} \phi_i \prod_{j=1}^{n} h(a_j; \mu_{i,j}, \sigma_{i,j}) . \tag{7}$$

Such a distribution can be sampled by drawing a component, individually sampling the component along each axis and squashing:

$$\begin{aligned} z &\sim \mathrm{Categorical}(\cdot; \boldsymbol{\phi}) \\ u_j &\sim \mathcal{N}(\cdot; \mu_{z,j}, \sigma_{z,j}) \\ \mathbf{a} &= \left[\tanh u_1 \quad \dots \quad \tanh u_n\right] \end{aligned}$$

However, the above approach is not differentiable regarding all parameters $\boldsymbol{\phi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. Thus, we propose rewriting the objective (2) such that all mixture components are sampled:

$$\begin{aligned} J(\pi) = \mathop{\mathbb{E}}_{s \sim \mathcal{D}} \Bigg[\sum_{i=1}^{k} \phi_i \mathop{\mathbb{E}}_{\mathbf{u} \sim \mathcal{N}(\cdot; \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)} [Q(s, \tanh \mathbf{u}) - \\ \alpha \log \pi(\tanh \mathbf{u}|s)] \Bigg] . \end{aligned} \tag{8}$$

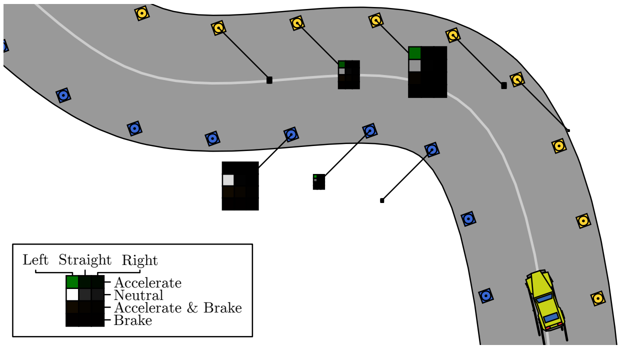

Fig. 3. Explanation of an agent with our explainable categorical mixture policy on the *Country Road* scenario using discrete actions. The explanation matrix encodes the probability of selecting one of the three longitudinal control inputs for steering and four lateral control inputs for accelerator and brake, which are activated independently. An explanation how to read the matrix is given in the box, where the intensity shows probability, thus black has no probability. The agent observes cones marking the boundary of the track. With our proposed architecture the influence on decision-making per object can be directly visualized. Most cones are ignored, as they have no attached explanation. The first cones following the left-hand curve are given high weight, as shown by the size of the attached explanations. The resulting combined probability distribution is the one shown in the box.

The sampling of $\mathbf{u} \sim \mathcal{N}(\cdot; \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ is realized using reparameterization:

$$\mathbf{u} = \boldsymbol{\mu}_i + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}_i, \text{ with } \boldsymbol{\epsilon} \sim \mathcal{N}(\cdot; \mathbf{0}, \mathbf{1}), \tag{9}$$

where $\odot$ is element-wise multiplication. The objective in (8) requires evaluation of Q k-times compared to once per sample in (2), since all components are individually sampled.

E. Regularization

To prevent the collapse of the mixture distribution, $z \sim \mathrm{Categorical}(\cdot; \boldsymbol{\phi})$ can optionally be entropy regularized. This can force the policy π to be more diverse in the attended components by adding J_R to the objectives (4) or (8):

$$J_R(\pi) = -\kappa \sum_{i=1}^{k} \phi_i \log(\phi_i) , \tag{10}$$

where κ is a hyper-parameter.

F. Network Architecture

In this section we briefly describe the network architectures used for the critic and our proposed explainable policy.

1) Critic: For the critic Q we use the previously suggested Deep Set with Learned Fourier Features architecture [2] for scenes with set-based observation spaces, and regular MLPs for ordered fixed-sized sequences, like lists of trajectory points, which is a common choice [3]. As mentioned in the previous section, differentiating the new objectives (4) and (8) requires evaluation of Q for the same observation s multiple times. In the discrete case, Q is realized as $Q : \mathcal{S} \to \mathbb{R}^n$, efficiently predicting a vector of all state-action values for a given state at once [10]. In the continuous case, Q first embeds s using a feature extractor F, then a critic C estimates the state-action value: $Q(s, a) = C(F(s), a)$. Thus by storing the intermediate result of $F(s)$, we can achieve a large gain of performance if F is computationally expensive compared to C.

2) Actor: An overview of our explainable policy architecture is given in Figure 2 with a comparison to baseline policies on set- or sequence-based observation spaces [2], [27]. In our proposed architecture, per sub-space of the observation, for example a single other vehicle or single trajectory point, the policy network estimates the parameters of the action distribution. Static information like vehicle odometry is concatenated to each input. Furthermore, if components of the sub-space are ordered, for example the observation space is a sequence, a positional encoding is concatenated. The policy is an MLP, which estimates the parameters ψ of a single mixture component and its component weight ϕ. We propose to optionally estimate ϕ from an earlier hidden layer than ψ, as indicated in Figure 2, which can substantially reduce the computational cost when sampling the policy. To train the policy in SAC, the default objective (2) is replaced by either (4) or (8) depending on action space, optionally adding the regularization from (10).

IV. ENVIRONMENT

We evaluate our proposed method on several driving scenarios using the *CarEnv* [2] simulation. This simulation uses a dynamic single-track vehicle model which makes control difficult. In all scenarios, vehicle odometry is given as $\mathbf{x}_{\text{odo}} = \begin{bmatrix} v_x & v_y & \beta & \omega & \omega_f & \omega_r \end{bmatrix}$, where (v_x, v_y) is the velocity in the vehicle reference frame, β the current steering angle, ω the yaw rate, and ω_f and ω_r the rotation speed of front and rear axle, respectively. The action space is given as $\mathbf{u} = \begin{bmatrix} u_s & u_a & u_b \end{bmatrix}$, where steering u_s, accelerator u_a and brake u_b are controlled independently. The following scenarios are used:

A. Country Road

This task was established in previous work [2]. The agent controls a vehicle following a track by observing road boundary markers, which mark the left and right edges. For an illustration of the environment, see Figure 3. The k dynamic objects are perceived as the set $\mathbf{x}_{\text{dynamic}} = \{\begin{pmatrix} x_i & y_i & \mathbb{1}_{\text{left},i} & \mathbb{1}_{\text{right},i} \end{pmatrix}\}_{i=1}^{k}$, where (x_i, y_i) is an object's position in vehicle reference frame and $\mathbb{1}_{\text{left},i}$ indicates that it is a left boundary marker, and $\mathbb{1}_{\text{right},i}$ that it is a right one. Static observation is the vehicle odometry: $\mathbf{x}_{\text{static}} = \mathbf{x}_{\text{odo}}$. The agent gains reward by navigating the track quickly and safely, without leaving the track surface. The reward is:

$$r(s,a) = -\mathbb{1}_{\text{fail}} - 0.2 \cdot \mathbb{1}_{\text{collision}} + 0.01 \cdot \mathbf{n}_{\text{track}} \cdot \mathbf{v}_{\text{ego}} \,, \quad (11)$$

where $\mathbb{1}_{\text{fail}}$ indicates leaving the track surface, immediately terminating the episode, $\mathbb{1}_{\text{collision}}$ indicates colliding with a boundary marker, $\mathbf{n}_{\text{track}}$ is the track direction at the projection of the vehicle, and $\mathbf{v}_{\text{ego}}$ is the velocity vector of the vehicle.

B. Tracking

A very common problem in automated driving involves the tracking of a reference trajectory using a learned controller [3]. As a recently proposed addition [28], we modify the *Country Road* scenario of *CarEnv* to use the centerline of the track as reference instead of boundary markers. As such the object observation is changed to an ordered sequence of k centerline points: $\mathbf{x}_{\text{dynamic}} = \left(\begin{pmatrix} p_{x,i} & p_{y,i} \end{pmatrix}\right)_{i=1}^{k}$, where $(p_{x,i}, p_{y,i})$ is the position of the i-th point in the vehicle reference frame. We use $k = 25$ points with a spacing of $2\,\text{m}$. See Figure 4 for an illustration.

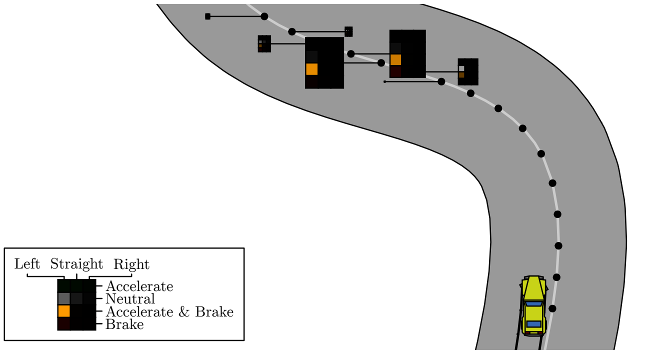

Fig. 4. Explanation of our explainable policy on the *Tracking* scenario. Here, the agent controls the vehicle observing the track centerline as an ordered sequence of points. The explanation is presented in the same way as in Figure 3.

C. Intersection

We also evaluate our proposed method on a novel multi-agent scenario for the *CarEnv* simulator, called *Intersection*. Here, multiple agents must learn to navigate an unregulated four-way intersection cooperatively. The agents have partial observability from their own view and do not share information, thus they operate decentralized. We implement this scenario using *CarEnv*'s dynamics model, making the problem more difficult compared to simulations with kinematics. The object observations of each agent are the set of other vehicles: $\mathbf{x}_{\text{dynamic}} = \{\begin{pmatrix} x_i & y_i & v_{x,i} & v_{y,i} & \cos\theta_i & \sin\theta_i \end{pmatrix}\}_{i=1}^{k}$, where (x_i, y_i) is the other vehicle's position in the vehicle reference frame, $(v_{x,i}, v_{y,i})$ the other vehicle's velocity vector in the vehicle reference frame and θ_i the orientation relative to the agent's vehicle's own orientation. Static observations are the ego-vehicle odometry and N navigation points: $\mathbf{x}_{\text{static}} = \mathbf{x}_{\text{odo}} || \begin{bmatrix} p_{x,1} & p_{y,1} & \dots & p_{x,N} & p_{y,N} \end{bmatrix}$, with $(p_{x,j}, p_{y,j})$ as the relative position of the j-th navigation point in vehicle reference frame. Theses navigation points indicate to the agent which exit of the intersection to use. Quickly following the centerline of the agent's lane is rewarded, while collisions are punished. Each agent receives a reward of 1 when it completes its route, 0 if it collides but is not moving, and -1 if it collides while moving. All these events remove it from the simulation. Otherwise, on non-terminal transitions, the reward is:

$$r(s,a) = k_1 \cdot \mathbf{n}_{\text{traj}} \cdot \mathbf{v}_{\text{ego}} - k_2 \cdot d_{\text{traj}} \cdot \|\mathbf{v}_{\text{ego}}\| \,, \quad (12)$$

where $\mathbf{n}_{\text{traj}}$ is the reference trajectory direction at the foot point of the ego vehicle, and d_{traj} the distance to the foot

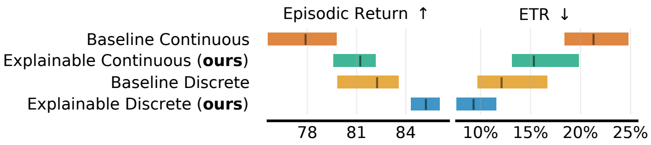

Fig. 5. Final performance on the *Country Road* scenario using continuous and discrete actions. Using discrete actions, our explainable policy is significantly better, achieving higher episodic return and lower early termination rate (ETR). For continuous actions performance is slightly better, but not significant at $p = 0.05$.

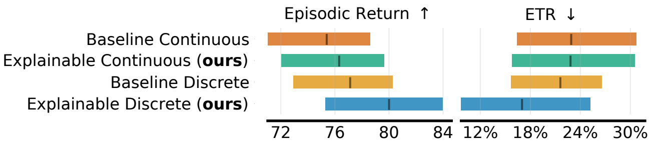

Fig. 6. Final performance on the *Tracking* scenario. Our proposed explainable policies on both continuous and discrete action spaces perform at least on-par with the baseline in both episodic return and early termination rate, with differences not significant at $p = 0.05$, while offering much better explainability.

point. Thus, the agent is rewarded for following the reference trajectory through the intersection while keeping the tracking error small. The coefficients $k_1 = 0.005$ and $k_2 = 0.002$ were chosen empirically.

V. Experiments

In this section, extensive experiments are performed to compare our explainable policy with a standard SAC architecture on both discrete and continuous action spaces, sequence- and set-based observation spaces as well as single agent and multi agent scenarios. Furthermore, the computational cost of our proposed method is analyzed. At first the training setup is described.

A. Experimental Setup

In all experiments, we train 20 agents per configuration with individual seeds. We evaluate their performance using bootstrapping to estimate interquartile mean (IQM) and its confidence intervals (CI) [29]. We report $95\,\%$ confidence intervals in tables and as shaded areas in graphs. Our experiments include agents with discrete and continuous action spaces. We discretize the continuous action space of $\mathbf{u} = \begin{bmatrix} u_s & u_a & u_b \end{bmatrix}$ individually with 3 steps for steering and 2 steps for accelerator and brake control. Thus, the discrete action space is given by $\mathbf{u}_{\text{discrete}} \in \{-1, 0, 1\} \times \{-1, 1\} \times \{-1, 1\}$. For single agent scenarios $1\,000\,000$ environment steps are used for training, a buffer size of $1\,000\,000$, batch size of 256, Polyak update step $\tau = 0.005$ (from [1]), learning rate $\mathrm{lr} = 0.0001$ and discount factor $\gamma = 0.95$ (from [2]). For multi agent scenarios $3\,000\,000$ environment steps are used, a buffer size of $8\,000\,000$ and batch size of 512. One gradient step is carried out per collected transition. Automatic entropy adjustment from [23] is used with a target entropy of $-n$ for continuous actions and $0.5 \cdot \log(n)$ for discrete actions. Hyper-parameters not taken from the literature were optimized using grid searches. For our explainable policy we always use an MLP with hidden dimensions $256 - 256 - 256$, where the mixture component weight is estimated from the final hidden layer unless stated otherwise, and set $\kappa = 0.001$ from (10).

B. Single Agent Performance

First, we evaluate the episodic return achieved on the single agent scenarios. On *Country Road*, for the baseline SAC policy and all critic networks, we use the Learned Fourier Feature Deep Set-based network architecture proposed by earlier work [2]. The integration of our proposed architecture is straight-forward as described in section III-F.2: per observed cone, a policy processes the cone encoding concatenated with ego-vehicle odometry. The results are shown in Figure 5. Our explainable policy performs on-par with the baseline on the continuous action space and significantly better than the baseline for discrete action spaces, achieving a higher episodic return of 85.22 compared to the baseline with 82.24. Early termination rate is also lowest for our explainable policy using discrete actions with $9.3\,\%$. This is achieved while also being explainable regarding the influence of individual objects on decision-making, as shown in Figure 3 and the supplementary video.

Next, we evaluate performance on the *Tracking* scenario, where the policy must solve the same problem using a different observation space, perceiving the track as a sequence of centerline points rather than a set of objects. Both for the baseline policies and the critic networks, an MLP is used as established in the literature on similar tasks [3], [28]. Hidden dimensions of $256 - 256 - 256$ are used, as determined by a grid search. For our explainable mixture policy, we want to explain decision-making per point of the reference as shown in Figure 4, thus we infer the explainable policy per reference point. As these points are ordered, we add a positional encoding to maintain ordering through the set of observations. The results of the experiment are shown in Figure 6. There are no statistically significant differences between any two methods. However, our explainable policy using discrete actions once again achieves good results with an episodic return of 80.00 and early termination rate of $17.0\,\%$, while also being more explainable. See Figure 4 and the supplementary video for examples.

C. Multi Agent Performance

Next, we evaluate the performance on the *Intersection* scenario (Figure 1), where multiple vehicles controlled in a decentralized fashion must cross an unregulated intersection. We use curriculum learning, where the number of vehicles is randomly selected from 1, 2, 4 and 8 and the simulation restarted after $60\,\mathrm{s}$ of simulated time. Furthermore, periodic re-initializing [30] of the neural networks is used every $200\,000$ steps. We employ a similar Deep Set architecture for the critic and baseline policies as used for *Country Road*. The performance in comparison to the baseline over the course of the training is shown in Figure 7. Using both discrete and continuous action spaces, our explainable policies are more sample efficient, completing significantly more trips

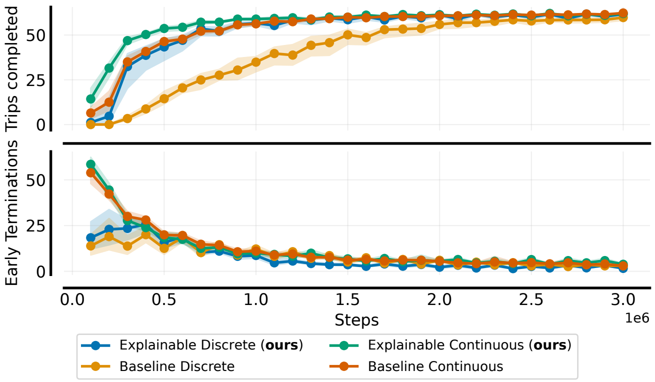

Fig. 7. Sample efficiency on the multi agent *Intersection* scenario. Our explainable policies are more sample efficient, achieving significantly more completed trips than the baselines at early stages of the training without sacrificing safety (early terminations), while also offering better explainability.

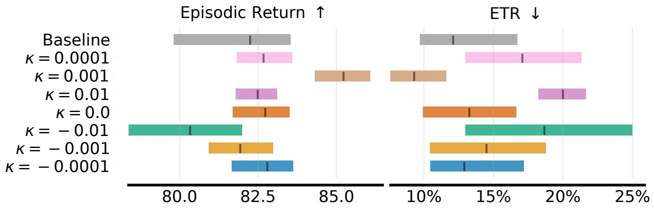

Fig. 8. Influence of different choices of the mixture regularization coefficient κ for our explainable policy on the final agent performance on the *Country Road* scenario with discrete action space. Baseline is the regular policy architecture.

early in the training. The discrete version of our explainable policy also has significantly more trip completions at the end of training with 61.68 (CI 95% [60.98, 62.04]) than the discrete baseline 59.77 (CI 95% [58.02, 60.47]) and is with significance the safest of all policies with 1.62 early terminations (CI 95% [1.22, 1.95]) compared to the discrete baseline with 3.04 early terminations (CI 95% [2.71, 3.52]). These good results are again attained while offering far more insights into the decision-making of the policy, see Figure 1 and the supplementary video for examples.

D. Analyses

Our architecture introduces a single hyper-parameter, κ, to regularize the entropy of the mixture components. The aim when using a positive kappa is to prevent quick convergence to a single component. The importance of κ and robustness of the algorithm regarding its choice are evaluated. Performance using different positive and negative values as well as no regularization on the discrete *Country Road* scenario are analyzed in Figure 8. A choice of $\kappa = 0.001$, which is the value used in all experiments, achieves the best episodic return with significance. However, within a reasonable range the algorithm is robust to the choice of κ, with no choice performing significantly worse than the regular policy baseline in terms of episodic return.

TABLE I

COMPARISON OF TRAINABLE PARAMETER COUNT, FLOATING POINT OPERATIONS (FLOPS), AND EPISODIC RETURN ON THE *Country Road* SCENARIO FOR BASELINE POLICY AND OUR PROPOSED POLICY DEPENDING ON HIDDEN LAYERS FOR COMPONENT WEIGHTS (ϕ_d).

Architecture	FLOPS	Params.	Episodic Return ↑	
			IQM	CI 95 %
Baseline	7 096 281	200 710	77.89	[75.59, 79.80]
Ours ($\phi_\mathrm{d} = 1$)	988 262	136 455	79.12	[77.28, 80.39]
Ours ($\phi_\mathrm{d} = 2$)	14 015 078	136 455	78.70	[76.26, 80.55]
Ours ($\phi_\mathrm{d} = 3$)	27 041 896	136 455	81.23	[79.58, 82.18]

E. Compute Efficiency

Finally, we analyze the efficiency of our proposed policies. As mentioned in Section III-F.2, we have proposed estimating the component weights from an earlier hidden layer than the mixture components for faster inference. We compare the number of trainable parameters and arithmetic operations between the baseline and our proposed architecture with different depths ϕ_d for the mixture component weights in Table I on the *Country Road* scenario using a continuous action space. The entire policy is an MLP with depth 3 and width 256 as in the main experiments. The depth at which the component weight is estimated has no influence on the number of parameters, as all hidden dimensions are identically sized. With fewer trainable parameters and far lower number of FLOPS, when estimating ϕ using a single hidden layer ($\phi_\mathrm{d} = 1$), our proposed policy still performs on-par with the baseline.

VI. CONCLUSIONS & FUTURE WORK

In this paper we have presented a novel formulation for reinforcement learning policies using mixture distributions which are naturally explainable. We have shown competitive performance compared to standard RL policies on different established scenarios in automotive applications using high-level representations, like object features and waypoints. For future research, we want to further investigate the application to different data modalities. For example, patches of an image may be suitable sub-spaces for camera observations, and we want to investigate application to slices of a lidar scan. While we have applied the mixture distributions to improve explainability, they also introduce multi-modality, for example a vehicle avoiding a head-on collision may be able to steer left or right. Mixture distributions may help remedy policies quickly converging to either choice of direction, which warrants further research. Finally, we have analyzed the application to SAC in this paper, an algorithm selected due to its importance to the robotics and AD communities. However, our explainable policies can be applied to any RL framework with stochastic policies, thus testing on on-policy methods such as PPO may also be an interesting direction for future research.

REFERENCES

[1] 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*, pp. 1861–1870, PMLR, 2018.

[2] M. Schier, C. Reinders, and B. Rosenhahn, "Learned fourier bases for deep set feature extractors in automotive reinforcement learning," in *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 931–938, IEEE, 2023.

[3] P. R. Wurman, S. Barrett, K. Kawamoto, J. MacGlashan, K. Subramanian, T. J. Walsh, R. Capobianco, A. Devlic, F. Eckert, F. Fuchs, *et al.*, "Outracing champion gran turismo drivers with deep reinforcement learning," *Nature*, vol. 602, no. 7896, pp. 223–228, 2022.

[4] Y. Lu, J. Fu, G. Tucker, X. Pan, E. Bronstein, R. Roelofs, B. Sapp, B. White, A. Faust, S. Whiteson, *et al.*, "Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7553–7560, IEEE, 2023.

[5] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[6] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information fusion*, vol. 58, pp. 82–115, 2020.

[7] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature communications*, vol. 10, no. 1, p. 1096, 2019.

[8] A. Atrey, K. Clary, and D. Jensen, "Exploratory not explanatory: Counterfactual analysis of saliency maps for deep reinforcement learning," in *International Conference on Learning Representations*, 2020.

[9] S. Milani, N. Topin, M. Veloso, and F. Fang, "Explainable reinforcement learning: A survey and comparative review," *ACM Computing Surveys*, vol. 56, no. 7, pp. 1–36, 2024.

[10] P. Christodoulou, "Soft actor-critic for discrete action settings," *arXiv preprint arXiv:1910.07207*, 2019.

[11] Y. Choi, K. Lee, and S. Oh, "Distributional deep reinforcement learning with a mixture of gaussians," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9791–9797, IEEE, 2019.

[12] B. Shahriari, A. Abdolmaleki, A. Byravan, A. Friesen, S. Liu, J. T. Springenberg, N. Heess, M. Hoffman, and M. Riedmiller, "Revisiting gaussian mixture critics in off-policy reinforcement learning: a sample-based approach," *arXiv preprint arXiv:2204.10256*, 2022.

[13] D. W. Nam, Y. Kim, and C. Y. Park, "Gmac: A distributional perspective on actor-critic framework," in *International Conference on Machine Learning*, pp. 7927–7936, PMLR, 2021.

[14] M.-S. Kim, G. Eoh, and T.-H. Park, "Decision making for self-driving vehicles in unexpected environments using efficient reinforcement learning methods," *Electronics*, vol. 11, no. 11, p. 1685, 2022.

[15] M. Sperl, L. Saluzzi, L. Grüne, and D. Kalise, "Separable approximations of optimal value functions under a decaying sensitivity assumption," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 259–264, IEEE, 2023.

[16] B. Peng, T. Rashid, C. Schroeder de Witt, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Facmac: Factored multi-agent centralised policy gradients," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12208–12221, 2021.

[17] R. Paleja, Y. Niu, A. Silva, C. Ritchie, S. Choi, and M. Gombolay, "Learning Interpretable, High-Performing Policies for Autonomous Driving," in *Proceedings of Robotics: Science and Systems*, 2022.

[18] Z. Song, Y. Jiang, J. Zhang, P. Weng, D. Li, W. Liu, and J. Hao, "An interpretable deep reinforcement learning approach to autonomous driving," in *IJCAI Workshop on Artificial Intelligence for Automous Driving*, 2022.

[19] S. Greydanus, A. Koul, J. Dodge, and A. Fern, "Visualizing and understanding atari agents," in *International conference on machine learning*, pp. 1792–1801, PMLR, 2018.

[20] A. Mott, D. Zoran, M. Chrzanowski, D. Wierstra, and D. Jimenez Rezende, "Towards interpretable reinforcement learning using attention augmented agents," *Advances in neural information processing systems*, vol. 32, 2019.

[21] L. Chen, O. Sinavski, J. Hünermann, A. Karnsund, A. J. Willmott, D. Birch, D. Maund, and J. Shotton, "Driving with llms: Fusing object-level vector modality for explainable autonomous driving," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14093–14100, IEEE, 2024.

[22] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[23] 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.

[24] B. G. Lindsay, "Mixture models: Theory, geometry and applications," in *NSF-CBMS Regional Conference Series in Probability and Statistics*, pp. i–163, JSTOR, 1995.

[25] N. Vieillard, O. Pietquin, and M. Geist, "Munchausen reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4235–4246, 2020.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[27] M. Huegle, G. Kalweit, B. Mirchevska, M. Werling, and J. Boedecker, "Dynamic input for deep reinforcement learning in autonomous driving," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 7566–7573, IEEE, 2019.

[28] E. Cramer, B. Frauenknecht, R. Sabirov, and S. Trimpe, "Contextualized hybrid ensemble q-learning: Learning fast with control priors," *arXiv preprint arXiv:2406.19768*, 2024.

[29] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare, "Deep reinforcement learning at the edge of the statistical precipice," *Advances in neural information processing systems*, vol. 34, pp. 29304–29320, 2021.

[30] E. Nikishin, M. Schwarzer, P. D'Oro, P.-L. Bacon, and A. Courville, "The primacy bias in deep reinforcement learning," in *International conference on machine learning*, pp. 16828–16847, PMLR, 2022.