Multimodal Safety-Critical Scenarios Generation for Decision-Making Algorithms Evaluation

Wenhao Ding, Baiming Chen, Bo Li, Kim Ji Eun, and Ding Zhao

***Abstract*—Existing neural network-based autonomous systems are shown to be vulnerable against adversarial attacks, therefore sophisticated evaluation of their robustness is of great importance. However, evaluating the robustness under the worst-case scenarios based on known attacks is not comprehensive, not to mention that some of them even rarely occur in the real world. Also, the distribution of safety-critical data is usually multimodal, while most traditional attacks and evaluation methods focus on a single modality. To solve the above challenges, we propose a flow-based multimodal safety-critical scenario generator for evaluating decision-making algorithms. The proposed generative model is optimized with weighted likelihood maximization and a gradient-based sampling procedure is integrated to improve the sampling efficiency. The safety-critical scenarios are generated by efficiently querying the task algorithms and a simulator. Experiments on a self-driving task demonstrate our advantages in terms of testing efficiency and multimodal modeling capability. We evaluate six Reinforcement Learning algorithms with our generated traffic scenarios and provide empirical conclusions about their robustness.**

***Index Terms*—Robot safety, reinforcement learning, semantic scene understanding.**

I. Introduction

ROBUSTNESS and safety are crucial factors to determine whether a decision-making algorithm can be deployed in the real world [1]. However, most of the data collected from simulations or in the wild are skewed to redundant and highly safe scenarios, which leads to the long tail problem [2]. Furthermore, a self-driving vehicle has to drive hundreds of millions of miles to collect safety-critical data [3], resulting in expensive development and evaluation phases. Meanwhile, a large number of safe Reinforcement Learning (RL) algorithms [4] have been proposed recently, yet the evaluation of these algorithms mostly use uniform sampling scenarios, which have been proven to be insufficient due to poor coverage of rare risky events.

Manuscript received October 15, 2020; accepted February 2, 2021. Date of publication February 16, 2021; date of current version March 2, 2021. This letter was recommended for publication by Associate Editor C. Gosselin and J. Zhang upon evaluation of the reviewers' comments. *(Corresponding author: Ding Zhao.)*

Wenhao Ding, Baiming Chen, and Ding Zhao are with the Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh 15206, PA USA (e-mail: wenhaod@andrew.cmu.edu; cbm17@mails.tsinghua.edu.cn; dingzhao@cmu.edu).

Bo Li is with the Department of Computer Science, UIUC, IL 61348 USA (e-mail: lbo@illinois.edu).

Kim Ji Eun is with the Robert Bosch LLC, PA USA (e-mail: jieun.kim@us.bosch.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3058873 provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3058873

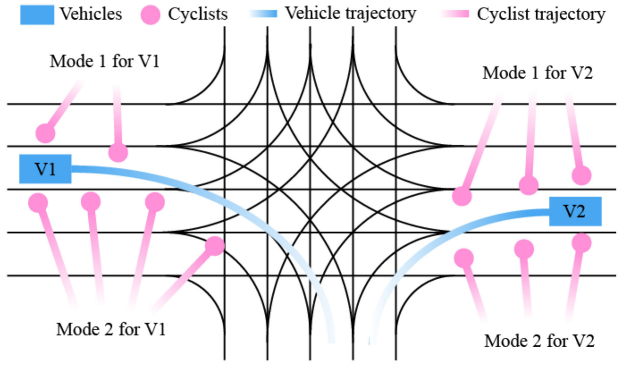

Fig. 1. Examples of multimodal safety-critical traffic scenarios. For each vehicle, there is a two-mode distribution of the cyclist's spawn point that leads to a high probability of collision. Pink trajectories represent samples from the distributions.

Adversarial attack [5], [6] is widely used to obtain specific examples when assessing the robustness of the model. This method only addresses extreme conditions, thus it does not provide comprehensive performance evaluations of the system. Researchers [7], [8] point out that there will always be loopholes in a neural network (NN) that can be attacked, hence, testing at different stress levels is deemed to provide more information about the robustness of the system. On the other hand, although the perturbation is limited during the attack, there is no guarantee that obtained samples are likely to occur in the real-world. It is a waste of resources to request robots to pass the tests that are unlikely to happen in practice.

The real-world scenarios are complicated with a huge number of parameters, and risk scenarios do not always happen within certain modality [9]. Multimodal distribution is a more realistic representation, for example, accidents could happen in different locations for a self-driving car as shown in Fig. 1. Though previous works [10], [11] tried to search the risk scenarios under the RL framework, they only use the single-mode Gaussian distribution policy, leading to the over-fitting and unstable training problems. Covering diverse testing cases provides a more accurate comparison of algorithms. Even if a robot overfits to one specific risk modality, it will fail to handle other potential risk scenarios. To the best of our knowledge, few people have explored the multimodal estimation of safety-critical data.

In this paper, we use a flow-based generative model to estimate the multimodal distribution of safety-critical scenarios. We use the weighted maximum likelihood estimation (WMLE) [12] as the objective function, where the weight is related to the risk

metric so that the log-likelihood of the sample will be approximately proportional to the risk level. We treat the algorithm that we want to evaluate as a black box, then get the risk value through the interaction with the simulation environment. To increase the generalization of generated scenarios, our generator also has a conditional input, so that the generated samples will be adaptively changed according to the characteristics of the task.

We model the whole training process as an on-policy optimization framework which shares the same spirit as Cross-Entropy Method (CEM) [13], and we propose an adaptive sampler to improve the sample efficiency. Under this framework, we can dynamically adjust the region of interest of the sampler according to the feedback of the generator. The adaptive process is guided by the gradient estimated from the Natural Evolution Strategy (NES) method [14]. During the training stage, the sampler focuses on the unexplored and risky areas, and finally completes the multimodal modeling. We also consider the distribution of real-world data when designing the metric of risk to ensure that the generated data has a high probability of occurrence in the real world.

We carried out extensive experiments on the decision-making task of autonomous driving in an intersection environment. A safety-critical generator is trained, analyzed, and compared with traditional methods. Our generator outperforms others in terms of efficiency and multimodal covering capability. We also evaluate the robustness of several RL algorithms and claim that our generator is more informative than uniform sampling methods. In summary, the contribution is three-fold:

1) We propose a multimodal flow-based generative model that can generate adaptive safety-critical data to efficiently evaluate decision-making algorithms.
2) We design an adaptive sampling method based on black-box gradient estimation to improve the sample efficiency of multimodal density estimation.
3) We evaluate a variety of RL algorithms with our generated scenarios and provide empirical conclusions that can help the design and development of safe autonomous agents.

II. Related Work

A. Deep Generative Model

Our method is based on deep generative models. The current popular generative models are mainly divided into three categories: normalizing flow [15] and autoregressive model [16] directly maximize the likelihood, Variational Auto-encoder (VAE) [17] optimizes the approximate likelihood using variational inference, and Generative adversarial network [18] implicitly computes the likelihood with a discriminator. The essence of these methods is to fit a distribution with parametric models that maximizes the likelihood according to the empirical data. In this paper, our data is collected from on-policy exploration. We select the flow-based model as the building block since we want to optimize the weighted likelihood directly and easily sample from the model.

B. Adversarial Attack

Another topic that is closely related to ours is the adversarial attack, which reduces the output accuracy of the target model by applying small disturbances to the original input samples. According to the information from the target model, this method falls into two types: white-box attack and black-box attack. Our method assumes that the internal information of the task algorithm cannot be obtained, so we are more relevant to the second one. This kind of method can be further divided into two mainstreams: substitute model [19] and query-based model [20]. The former is to train a completely accessible surrogate model to replace the target model, while the latter is based on the query of the target model to estimate the optimal attack direction. The NES gradient estimation method used in this paper has shown promising results in the latter method [21].

C. Safety-Critical Scenario Generation

Some previous works have used the generative model to conduct safety-critical scenarios search. [22] modifies the last layer of a generative adversarial imitation learning model to generate different driving behaviors. [23] and [24] generate different levels of risky data by controlling the latent code of VAE. Besides, some frameworks [10], [11], [25] combine RL and simulation environment to search for data that satisfies specific requirements. However, they only consider single-modal Gaussian policy. Adaptive stress test [26], [27] is also a kind of methods using the Monte Carlo Tree Search and RL to generate collision scenarios. Most of these methods use the Gaussian distribution policy to describe the result of searching, without considering the case of multiple modalities. In addition, lots of literatures borrow the idea from evolution algorithms [28], reinforcement learning [29], Bayesian optimization [30], and importance sampling [31] to generate adversarial complex scenarios, resulting in diverse directions and platforms. In previous works, [32] is the most similar to this paper. They use the multilevel splitting method [33] to gradually extract risk scenarios by squeezing the searching area. However, it is sensitive to level partition and the efficiency of the Monte Carlo Markov Chain (MCMC) method is limited by query times and data dimension.

D. Sampling Methods

In online decision-making, especially RL tasks, exploration has always been a popular topic. The adaptive sampler proposed in this paper can also be categorized into this field. For the simple multi-arm bandit problem, traditional solutions are Upper Confidence Bound (UCB) [34] and Thompson sampling [35]. Recently, exploration methods based on curiosity [36], and information gain [37] also cause much attention. Also, works like [38] are based on the disagreement of ensemble models. To some extent, all these methods aim at modeling the environment and the explored area, to guide the sampler with the desired direction.

The gradient information usually facilities samplers faster convergence. Hamiltonian Monte Carlo (HMC) is a variant of the MCMC method that is more efficient than vanilla random walk counterparts. Motivated by this, our proposed adaptive sampler also uses a black-box estimator to obtain the gradient of a non-differential target function.

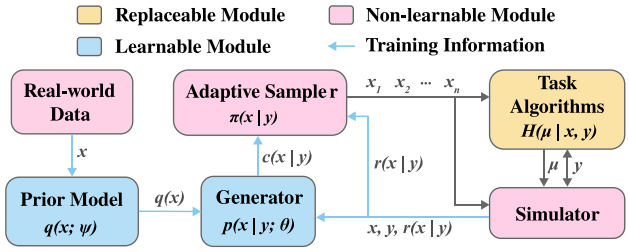

Fig. 2. Diagram of proposed framework. The modules that have learnable parameters are shown in blue.

III. Proposed Method

A. Notation and Preliminaries

The variable $\boldsymbol{x} \in \mathcal{X}$ represents parameters that build a scenario, for instance, the initial position of a pedestrian or the weather conditions in the self-driving context. The variable $\boldsymbol{y} \in \mathcal{Y}$ represents the properties of the task, such as the goal position or target velocity. $H(\boldsymbol{\mu}|\boldsymbol{x}, \boldsymbol{y})$ is the task algorithm that takes the scenario observation and task condition as input and outputs a decision policy $\boldsymbol{\mu} \in \mathcal{M}$. With a risk metric $r : \mathcal{X} \times \mathcal{M} \to \mathcal{R}$, each scenario is corresponding to a value that indicates the safety under H. For simplification, we omit the notation $\boldsymbol{\mu}$ in $r(\boldsymbol{x}, \boldsymbol{\mu}|\boldsymbol{y})$.

Instead of exploring the extreme scenarios that output low $r(\boldsymbol{x}|\boldsymbol{y})$ as in adversarial attack field, we aim at estimating a multimodal distribution $p(\boldsymbol{x}|\boldsymbol{y})$ where the log-likelihood is proportional to the value of risk measure. Then we can efficiently generate scenarios that have different risk levels by sampling from $p(\boldsymbol{x}|\boldsymbol{y})$. The condition $\boldsymbol{y}$ follows a distribution $p(\boldsymbol{y})$. A sampler $\pi(\boldsymbol{x}|\boldsymbol{y})$ is used to collect training data. We also assume real-world data of similar scenarios is accessible and has a distribution $q(\boldsymbol{x})$. The pipeline of our proposed evaluation method is shown in Fig. 2. We explain the details of three learnable modules in the following sections.

B. Pre-Trained Prior Model

Firstly, we consider the probability that each scenario happens in real-world to make the result practical. For that, we pre-train a generative model $q(\boldsymbol{x}; \boldsymbol{\psi})$ to approximate the distribution of the real data $\mathcal{D}$. The objective of the training is maximizing the following log-likelihood:

$$\hat{\boldsymbol{\psi}} = \arg\max_{\boldsymbol{\psi}} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \log q(\boldsymbol{x}; \boldsymbol{\psi}) \tag{1}$$

We select RealNVP [39], a flow-based model to implement $q(\boldsymbol{x}; \boldsymbol{\psi})$ for exact likelihood inference. In flow-based model, a simple distribution $q(\boldsymbol{z})$ is transformed to a complex distribution $p(\boldsymbol{x})$ by the change of variable theorem. Suppose we choose $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$ to be the simple distribution. Then we have the following equation to calculate the exact likelihood of $\boldsymbol{x}$:

$$q(\boldsymbol{x}) = q(\boldsymbol{z}) \left| \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} \right| = p(f(\boldsymbol{x})) \left| \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} \right| \tag{2}$$

where we have an invertible mapping $f : \mathcal{X} \to \mathcal{Z}$. For more details about the flow-based model, please refer to [39].

After training, scenarios can be generated by $\boldsymbol{x} = f^{-1}(\boldsymbol{z})$ where z is sampled from $\mathcal{N}(\mathbf{0}, \sigma)$. A smaller σ will make the samples more concentrated, therefore generated scenarios will have a higher likelihood. Since our model is trained with WMLE, the high likelihood samples are also corresponding to high risk. Details about the network structure and hyperparameters can be found in the Appendix.

C. Generator

We formulate the safety-critical data generation as a density estimation problem. The traditional way to estimate the multimodal distribution $p(\boldsymbol{x}|\boldsymbol{y})$ by a deep generative model is maximizing the likelihood of data. To integrate the risk information, we solve the problem by WMLE [12]. For one data point $\boldsymbol{x}_i$, we have:

$$L(\boldsymbol{x}_i|\boldsymbol{y}_i; \boldsymbol{\theta}) = p(\boldsymbol{x}_i|\boldsymbol{y}_i; \boldsymbol{\theta})^{w(\boldsymbol{x}_i)} \tag{3}$$

$$\log L(\boldsymbol{x}_i|\boldsymbol{y}_i; \boldsymbol{\theta}) = w(\boldsymbol{x}_i|\boldsymbol{y}) \log p(\boldsymbol{x}_i|\boldsymbol{y}_i; \boldsymbol{\theta}) \tag{4}$$

where $w(\boldsymbol{x}_i)$ is the weight and $p(\boldsymbol{x}_i|\boldsymbol{y}_i; \boldsymbol{\theta})$ is our generator with learnable parameter $\boldsymbol{\theta}$, corresponding to the i-th data point. Assume we have a sampling distribution $\pi(\boldsymbol{x}|\boldsymbol{y})$ of $\boldsymbol{x}$, then our objective is:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}\sim\pi(\boldsymbol{x}|\boldsymbol{y}), \boldsymbol{y}\sim p(\boldsymbol{y})} \log L(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta}) \tag{5}$$

The definition of $w(\boldsymbol{x}|\boldsymbol{y})$ is relevant to both $r(\boldsymbol{x}|\boldsymbol{y})$ and $q(\boldsymbol{x}; \boldsymbol{\psi})$:

$$w(\boldsymbol{x}_i) = r(\boldsymbol{x}_i|\boldsymbol{y}) + \beta q(\boldsymbol{x}_i; \boldsymbol{\psi}) \tag{6}$$

where β is a hyperparameter to balance $r(\boldsymbol{x})$ and $q(\boldsymbol{x}; \boldsymbol{\psi})$.

We implement $p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$ by a modified flow-based model that has a conditional input [40]. Suppose $\boldsymbol{y} \in \mathcal{Y}$ is the conditional input, then the mapping function should be $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ and (2) will be rewritten as:

$$p(\boldsymbol{x}|\boldsymbol{y}) = p(f(\boldsymbol{x}|\boldsymbol{y})) \left| \frac{\partial f(\boldsymbol{x}|\boldsymbol{y})}{\partial \boldsymbol{x}} \right| \tag{7}$$

D. Adaptive Sampler

The uniform distribution is a trivial choice for $\pi(\boldsymbol{x}|\boldsymbol{y})$ in (5) to search the solution space. However, uniform sampling is inefficient in high-dimensional space, especially when the risky scenarios are rare. Therefore, we propose an adaptive sampler that leverages the gradient information to gradually cover all modes of risky scenarios. Suppose we have a metric $c(\boldsymbol{x}|\boldsymbol{y})$ that indicates the exploration value: the higher $c(\boldsymbol{x}|\boldsymbol{y})$ is, the more worth exploring $\boldsymbol{x}$ is. We then use NES, a black-box optimization method, to estimate the gradient of the $c(\boldsymbol{x}|\boldsymbol{y})$. The sampler $\pi(\boldsymbol{x}|\boldsymbol{y})$ then follow the generating rule (we omit $\boldsymbol{y}$ in gradient derivation):

$$\boldsymbol{x}^{t+1} \leftarrow \boldsymbol{x}^t + \alpha \nabla_{\boldsymbol{x}} c(\boldsymbol{x}^t) \tag{8}$$

where α is the step size. The gradient $\nabla_{\boldsymbol{x}} c(\boldsymbol{x}^t)$ in (8) can be estimated by:

$$\begin{aligned} \nabla_{\boldsymbol{x}} c(\boldsymbol{x}^t) &= \nabla_{\boldsymbol{x}} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{x}^t, \sigma^2 \boldsymbol{I})} \left[c(\boldsymbol{x}) \right] \\ &= \frac{1}{\sigma} \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0}, \boldsymbol{I})} \left[\epsilon \cdot c(\boldsymbol{x}^t + \sigma\epsilon) \right] \end{aligned} \tag{9}$$

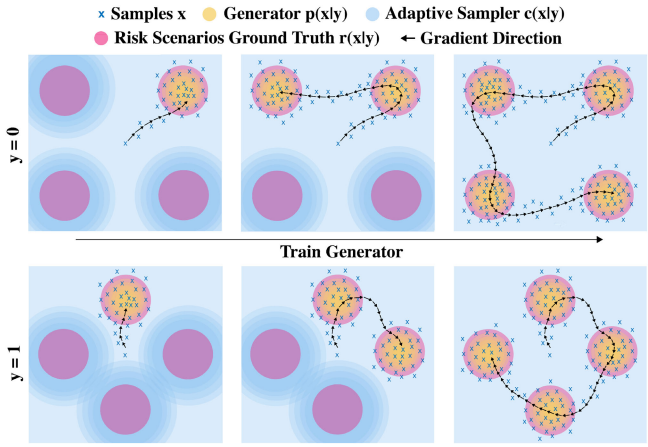

Fig. 3. A GMM example to illustrate our method. The condition $\boldsymbol{y}$ has two values to simulate conditional distribution.The ground truth $r(\boldsymbol{x}|\boldsymbol{y})$ is denoted by pink color, which is the likelihood of a 4-mode Gaussian distribution for each condition. The exploration value $c(\boldsymbol{x})$ is denoted by blue color, which gradually reduces as the samples expand. The generator $p(\boldsymbol{x})$ is denoted by yellow color, which gradually covers all modalities.

In practice, we will approximate the above expectation with the Monte Carlo method:

$$\nabla_{\boldsymbol{x}} c(\boldsymbol{x}^t) = \frac{1}{\sigma} \sum_{i=1}^{M} \epsilon_i \cdot c(\boldsymbol{x}^t + \sigma \epsilon_i), \quad \epsilon_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{10}$$

The design of $c(\cdot)$ heavily influences the performance of the adaptive sampler. Inspired by some curiosity-driven literature [36], where the uncertainty, Bayesian surprise, and prediction error are used to guide the exploration, we choose a metric that involves the generative model $p(\boldsymbol{x}|\boldsymbol{y})$:

$$c(\boldsymbol{x}|\boldsymbol{y}) = r(\boldsymbol{x}|\boldsymbol{y}) - \gamma \cdot p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta}) \tag{11}$$

where γ is a hyperparameter that balance $r(\boldsymbol{x})$ and $p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$. Intuitively, when one mode (some similar risky scenarios) is well learned by $p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$, the metric $c(\boldsymbol{x})$ will decrease and force the sampler to explore other modes. Finally, the multimodal distribution will be captured by $p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$. The diagram of this pipeline is shown in Fig. 3 with a Gaussian Mixture Model (GMM) example.

IV. Experiments

In this section, we firstly demonstrate the advantage of our proposed adaptive sampler with a toy example. After that, we show the generated safety-critical scenarios with different settings in an intersection environment. Finally, we evaluate the robustness of several popular RL algorithms using our generated scenarios and provide conclusions about their robustness.

A. Efficiency of Adaptive Sampler

Environment settings: As discussed in Section III-C, we shall expect that the estimated gradient of $c(\boldsymbol{x}|\boldsymbol{y})$ improves the efficiency of the sampling procedure. To assess this, we compare our method with two baselines in a similar GMM example to Fig. 3 under $y = 0$. In the first baseline, we use $c(\boldsymbol{x}) = r(\boldsymbol{x})$, a straightforward and common choice in the adversarial attack literature. In the second baseline, we use the variance of the posterior of Gaussian Processes (GP) to model the uncertainty of search space and combine this uncertainty with $r(\boldsymbol{x})$.

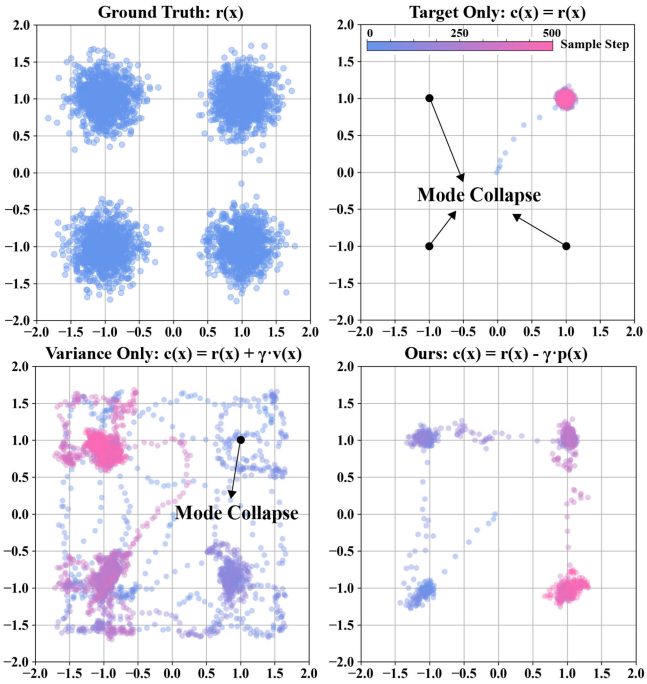

Fig. 4. A toy example to compare different adaptive samplers. The left-most figure shows the samples from $r(\boldsymbol{x})$, which is multimodal with four modalities. The other three figures show the samples obtained from three different samplers. Blue points indicate the early exploration stage and pink points indicate the later stage.

Explanation: The comparison results is displayed in Fig. 4. Both two baselines are facing the mode collapse problem to varying degrees, while our method effectively covers all modes. The reasoning is as follows. The first baseline uses only limited information about the multimodal landscape, thus is easily trapped into one modality. The second baseline, which gradually decreases the importance of the explored points, can cover all modes even other unimportant points. However, the rapidly descending uncertainty and the lack of adaptivity to the generator $p(\boldsymbol{x})$ lead to suboptimal results and unbalanced data collection. Our proposed method uses the feedback of the generator that gives the sampler both the capability of uncertainty exploration and balanced data collection, hence attaining all the modalities.

B. Safety-Critical Scenario Generation

Environment settings: An intersection environment is used to conduct our experiment in Carla simulator [41]. We represent the scenario as a 4-dimensional vector $\boldsymbol{x} = [x^s, y^s, v_x^s, v_y^s]$, which represents the initial position and initial velocity of a cyclist. The cyclist is spawned in the environment and travels at a constant speed. Then we place an ego vehicle controlled by an intelligent driver model. The target route of the ego vehicle is represented by condition $\boldsymbol{y}$. The minimal distance between the cyclist and ego vehicle is used to calculate $r(\boldsymbol{x})$:

$$r(\boldsymbol{x}) = \exp\{-\|\boldsymbol{p}_v - \boldsymbol{p}_c\|_2\} \tag{12}$$

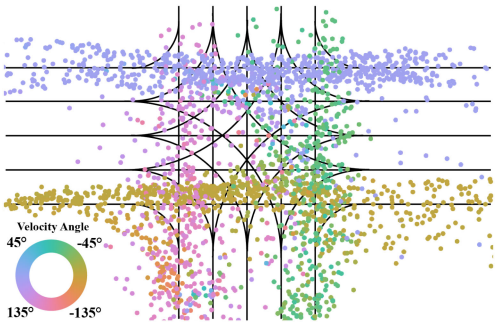

Fig. 5. Samples from prior model $q(\boldsymbol{x})$ that is learned from InD dataset [44]. Different colors represent different velocity angles.

where $\boldsymbol{p}_v$ and $\boldsymbol{p}_c$ represents the position of the vehicle and the cyclist respectively. A lower distance corresponds to a higher $r(\boldsymbol{x})$. This scenario is defined as a pre-crash scenario in [42] and also adopted in 2019 Carla Autonomous Driving Challenge [43]. This setting allows us to test the collision avoidance capability of decision-making algorithms H. Other more intelligent algorithms can replace the current agent during the evaluation stage.

Real-world data distribution: There are numerous datasets collected in the intersection traffic environment. We train our prior model $q(\boldsymbol{x})$ with trajectories from the InD dataset [44]. A well-trained prior model can be used to infer the likelihood of a given sample and generate new samples as well. We display the position and velocity direction of some generated samples in Fig. 5. These samples roughly describe the distribution of a cyclist in an intersection.

Generated scenarios display: We train a generator $p(\boldsymbol{x}|\boldsymbol{y})$ to generate safety-critical scenarios given the route condition $\boldsymbol{y}$. In Fig. 6, we compare the samples from two generators: one does not use prior (middle row) and the other uses $q(\boldsymbol{x})$ as the prior model (bottom row), where the same color map is used as in Fig. 5. The top row of Fig. 6 shows the collected scenarios by our adaptive sampler. Each of these samples is corresponding to a risk value that is not shown in the figure. In Fig. 6, it is shown that without real-world data prior, the generator learns the distribution of all risky scenarios collected by the adaptive sampler (first row). After incorporating the prior model (samples are shown in Fig. 5), the generator concentrates more on the samples that are more likely to happen in the real world. This results in the removal of samples that has opposite directions to the real data.

Baseline and metric settings: We select seven algorithms as our baseline. The details of each algorithm are discussed below:

1) Grid Search: We set the searching step for all parameters to 100. Since $\boldsymbol{x}$ has four dimensions, the entire searching iterations should be 10^8.
2) Human Design: We use the rules defined in https://github.com/carla-simulator/scenario_runnerCarla AD Challenge, which trigger the movement of the cyclist according to the location of the ego vehicle.
3) Uniform Sampling: The scenario parameter $\boldsymbol{x}$ is uniformly sampled from the entire space. This method is widely used in the evaluation of safety decision-making algorithms. For instance, obstacles are randomly generated to test the collision avoidance performance in the Safety-Gym environment [45].
4) REINFORCE [11]: This method uses the REINFORCE framework with a single Gaussian distribution policy. This kind of policy can only represent a single modality.
5) REINFORCE+GMM: The policy distribution of [11] is replaced with a GMM. The purpose is to explore the multimodal capability of the REINFORCE algorithm.
6) Ours-Uniform: We replace the adaptive sampler in our method with a uniform sampler.
7) Ours-HMC: We replace the adaptive sampler in our method with an HMC sampler to explore the efficiency of the gradient-based MCMC method.

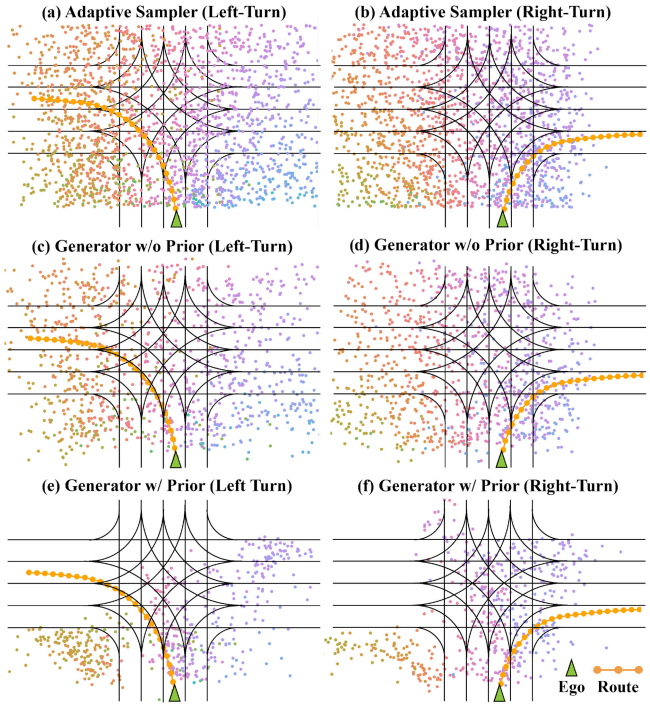

Fig. 6. Each point represents one risky scenario $\boldsymbol{x} = [x^s, y^s, v_x^s, v_y^s]$. The color represents the direction of the velocity (same as Fig. 5). Condition $\boldsymbol{y}$ represents the orange route.

We use the query time and collision rate as our metrics. The query time means the number of queries to the simulation during the training stage. Methods without training have 0 query time. A rough value is recorded when the distribution of samples is stable. The second metric is the collision rate, which is calculated after the training stage. We sample 1000 scenarios for 10 different routes and get a collision rate for each route. We then calculate the mean and variance across the 10 routes.

Comparison with baseline methods: The results are shown in Table. I. Grid search is the most trivial way comparable with our method in finding the multimodal risk scenarios. However, the query time grows exponentially as the dimension of $\boldsymbol{x}$ and the step size increase. In the simulation, human design is a possible way to reproduce the risk scenarios that happen in the real world, but these scenarios are fixed and not adaptive to the changes of the task parameters $\boldsymbol{y}$, leading to a low collision rate. Our experiment also found that the uniform method attains less than

TABLE I
PERFORMANCE COMPARISON

Methods	Queries (↓)	Collision Rate (↑)
Grid Search	1×10^8	$\mathbf{100\%}$
Human Design	-	$35\% \pm 21\%$
Uniform Sampling	-	$9\% \pm 1\%$
REINFORCE-Single [11]	$\mathbf{1 \times 10^3}$	$97\% \pm 2\%$
REINFORCE-GMM	$\mathbf{1 \times 10^3}$	$98\% \pm 1\%$
Ours-Uniform	1×10^5	$\mathbf{100\%}$
Ours-HMC	$\mathbf{1 \times 10^3}$	$\mathbf{100\%}$
Ours-Adaptive	3×10^3	$\mathbf{100\%}$

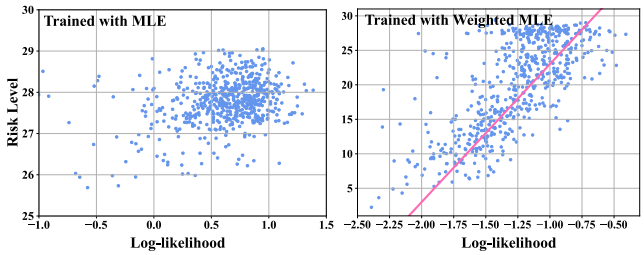

Fig. 7. Relationship between risk and log-likelihood of $p(\boldsymbol{x})$.

10% collision rate. In dense reward situations, uniform sampling could be a good choice, while in most real-world cases, the rare events risk scenarios makes this method quite inefficient. REINFORCE-Single searches the risk scenario under RL framework [11]. Although this method converges faster than ours, it cannot handle the multimodal cases with a single Gaussian distribution policy. The REINFORCE-GMM method extends the original version with a multimodal policy module. However, it has similar results as REINFORCE-Single. The reason is that the on-policy sampling method in REINFORCE is easy to be trapped into a single modality, even the policy itself is multimodal. The final weight in GMM is highly imbalanced and only one component dominates. The ablation study reveals that our adaptive sampler (Ours-Adaptive) is more efficient than the uniform version (Ours-Uniform). The MCMC version (Ours-HMC) requires less query time than our adaptive sampler, while its samples only concentrate on one modality.

Relationship between risk level and log-likelihood: Since our generator is trained with WMLE, we make usage of all collected samples rather than only the risky ones as in [32]. We compare two generators that are trained with MLE and WMLE and plot the results in Fig. 7. The generator trained with MLE by only using the risk data concentrates on the high-risk area, while our WMLE generator has a linear relationship between the risk and log-likelihood. Therefore, our generator can not only generate risky scenarios but also generate scenarios with different risk levels by considering the likelihood of samples.

C. Evaluation of RL Algorithms

To prove that our generated scenarios help improve the evaluation of algorithms, we implemented six popular RL agents (DQN [46], A2C [47], PPO [48], DDPG [49], SAC [50], Model-based RL [51]) as $H(\mu|\boldsymbol{x}, \boldsymbol{y})$ on the navigation task in the aforementioned environment. The target of agent is to arrive at a goal point $[x^g, y^g]$ and avoid reaching the non-driving area. At the same time, we place a cyclist on the intersection to create a traffic scenario. Finally, the state of the agent is:

$$s = [x^g, y^g, x^a, y^a, v_x^a, v_y^a, x^s, y^s, v_x^s, v_y^s] \tag{13}$$

where $[x^a, y^a, v_x^a, v_y^a]$ and $[x^s, y^s, v_x^s, v_y^s]$ represents the position and velocity of the agent and the cyclist, respectively. The agents should also avoid colliding the cyclist otherwise they will receive a penalty. The reward consists of three parts:

$$R(\boldsymbol{x}) = r_g + r_s \times I_s(\boldsymbol{x}) + r_c \times I_c(\boldsymbol{x}) \tag{14}$$

where r_g is calculated by reduction of distance between the agent and the goal, r_s and r_c indicates the penalty of non-driving area violation and cyclist collision. $I_s(\boldsymbol{x})$ and $I_c(\boldsymbol{x})$ are two indicator functions that equal to 1 when the two events happen. The episode terminates when the agent collides into the cyclist or the agent reaches the target. We implement DQN and A2C on discrete action space with a controller that follows a pre-defined route. Their action space only influences the acceleration. The other agents have continuous action space that controls throttle and steering.

We have two environments for training and testing: 1) Uniform Risk Scenarios (URS): the initial state $\boldsymbol{x}$ of the cyclist is uniformly sampled; 2) Generated Risk Scenarios (GRS): the initial state $\boldsymbol{x}$ is sampled from our generated $p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$ with $\sigma = 0.2$. We train and test six RL algorithms with different environments and Fig. 8 displays the testing reward and testing collision rate. According to the comparison between different settings, we draw three main conclusions:

More informative evaluation (Column 1 v.s. Column 2): Agents tested on URS have similar final rewards and collision rates. These nearly indistinguishable results make it difficult to compare the robustness of different algorithms. In contrast, the results on GRS show a great discrepancy, which helps us obtain clearer conclusions.

Generalization of agents (Column 1 v.s. Column 3): We train the agents on URS and GRS but test them both on URS. We notice that the performance of both settings are similar, which means all agents do not sacrifice their generalization to URS.

Robustness of different agents (Column 2 v.s. Column 4): The expected results should be that all agents have an improvement in column 4. However, we notice that different algorithms still show different robustness due to their mechanisms. We roughly divide the six algorithms into three categories and explain them respectively. 1) *Improved a lot:* MBRL is robust to the risk scenario, even if it is not trained on GRS. The reason is that the target of MBRL is to learn a dynamics model and plan with it. Even if it is trained on normal scenarios, it learns how to predict the trajectory of the cyclist. Then when it is tested on risky scenarios, MBRL can easily avoid the collision. Training on GRS will not make it perform better. 2) *Improved a little:* DDPG, DQN and SAC slightly improve the performance. From the bottom figure of column 4, we notice the collision rates of them firstly increase but quickly decrease, which means they learn how to deal with most of the risk scenarios. However, their rewards are lower than MBRL because they cannot handle all risky scenarios. The explanation for the little improvement

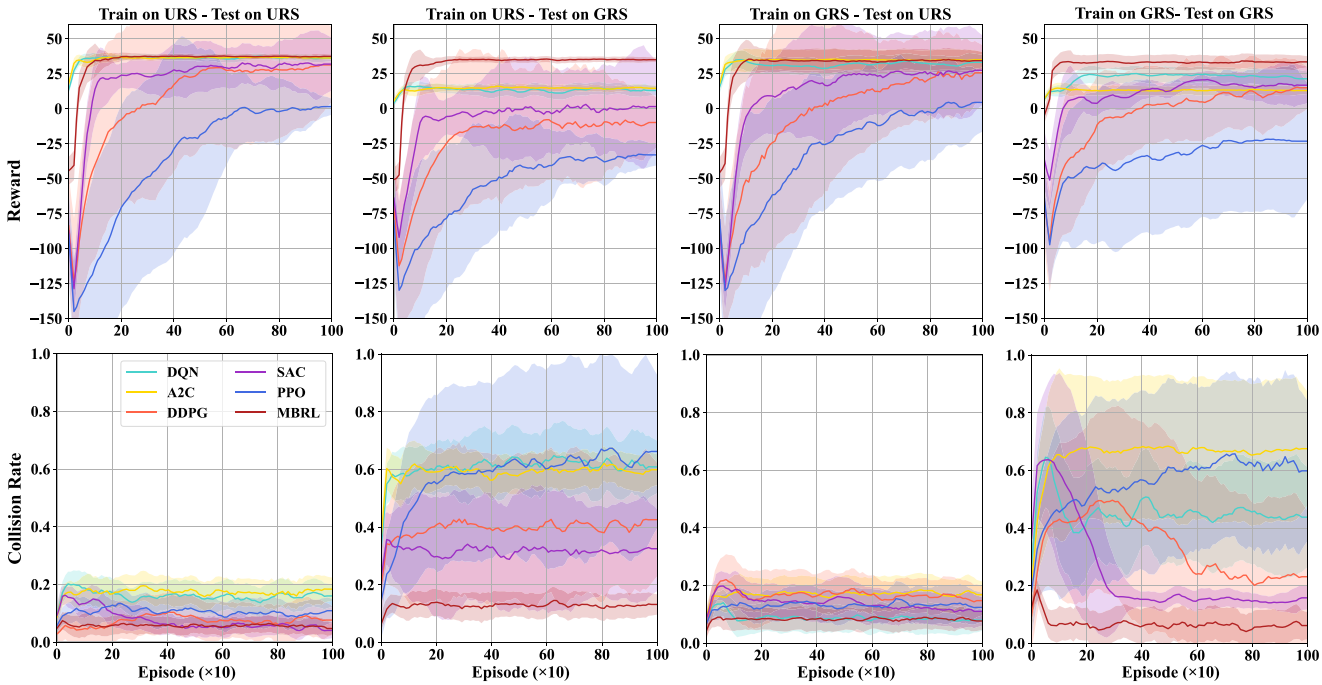

Fig. 8. Testing reward and testing collision rate in four different settings. Note that the action space of DQN and A2C is different from others, thus their results should not be compared to other methods. They are displayed together because they share the same reward space.

is that these are off-policy methods with memory buffers. The average of stored risky scenarios from the buffer makes the training stable, therefore makes the agents successfully handle the scenarios they have met. However, they still fail in some unseen risky scenarios. 3) *Not Improved:* PPO and A2C are on-policy methods, which learn policy according to current samples. However, the risky scenarios cause the instability of training, because our generator finds different modes (types) of risky scenarios. In contrast, a normal scenario will not cause such a problem because the state of the cyclist does not have much influence. In column 4, the collision rates of PPO and A2C gradually increase and never decrease, which means they cannot handle risky scenarios.

Note that the above empirical conclusions might only be valid in this environment. Further comparison of these RL algorithms should be carefully designed in multiple other settings. Nevertheless, our generator indeed is proven to be more insightful than the uniform sampler. Beyond RL algorithms, our proposed generating framework can also be used to efficiently evaluate other decision-making methods that are developed for dealing with more risky scenarios.

V. Conclusion

In this letter, we train a flow-based generative model using the objective function of weighted likelihood to realize the generation of multimodal safety-critical scenarios. Our generator can generate scenarios with various risk levels, providing efficient and diverse evaluations of decision-making algorithms. To speed up the training process, we propose an adaptive sampler based on a feedback mechanism, which can adjust the sampling region according to the learning progress of the generator, and finally cover all risk modes in a faster way. We test six RL algorithms with scenarios generated by our generator in a navigation task and obtain some conclusions that are not easy to get with traditional uniform sampling evaluation. This achievement provides an efficient evaluation and comparison test-bed for the safety decision-making algorithms which have recently attracted more and more attention. A potential extension of this work is combining the evaluation and training process to build an adversarial training framework. We expect this combination can boost existing algorithms under safety-related tasks.

Acknowledgment

The authors would like to thank Mansur Arief for helpful comments and discussion on drafts of this letter. This research was sponsored in part by Bosch.

References

[1] J. Fryman and B. Matthias, "Safety of industrial robots: From conventional to collaborative applications," in *Proc. ROBOTIK 2012; 7th German Conf. Robot. VDE*, 2012, pp. 1–5.

[2] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9268–9277.

[3] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?" *Transp. Res. Part A: Policy Pract.*, vol. 94, pp. 182–193, 2016.

[4] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, 2015.

[5] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 15–26.

[6] A. Nazemi and P. Fieguth, "Potential adversarial samples for white-box attacks," 2019, *arXiv:1912.06409*.

[7] A. Fawzi, H. Fawzi, and O. Fawzi, "Adversarial vulnerability for any classifier," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1178–1187.

[8] A. Shafahi, W. R. Huang, C. Studer, S. Feizi, and T. Goldstein, "Are adversarial examples inevitable?" 2018, *arXiv:1809.02104*.

[9] M. Yan, A. Li, M. Kalakrishnan, and P. Pastor, "Learning probabilistic multi-modal actor models for vision-based robotic grasping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 4804–4810.

[10] S. Kuutti, S. Fallah, and R. Bowden, "Training adversarial agents to exploit weaknesses in deep control policies," 2020, *arXiv:2002.12078*.

[11] W. Ding, M. Xu, and D. Zhao, "Learning to collide: An adaptive safety-critical scenarios generating method," 2020, *arXiv:2003.01197*.

[12] S. X. Wang, "Maximum weighted likelihood estimation," Ph.D. dissertation, University of British Columbia, 2001.

[13] R. Y. Rubinstein and D. P. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer Science & Business Media, 2013.

[14] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber, "Natural evolution strategies," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 3381–3387.

[15] D. P. Kingma and P. Dhariwal , "Glow: Generative flow with invertible 1x1 convolutions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10 215–10 224.

[16] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," 2013, *arXiv:1601.06759*.

[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, *arXiv:1312.6114*.

[18] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[19] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2017, pp. 506–519.

[20] C. Guo, J. R. Gardner, Y. You, A. G. Wilson, and K. Q. Weinberger, "Simple black-box adversarial attacks," 2019, *arXiv:1905.07121*.

[21] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," 2018, *arXiv:1804.08598*.

[22] M. O'Kelly, A. Sinha, H. Namkoong, R. Tedrake, and J. C. Duchi, "Scalable end-to-end autonomous vehicle testing via rare-event simulation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9827–9838.

[23] W. Ding, W. Wang, and D. Zhao, "A new multi-vehicle trajectory generator to simulate vehicle-to-vehicle encounters," 2018, *arXiv:1809.05680*.

[24] W. Ding, M. Xu, and D. Zhao, "Cmts: Conditional multiple trajectory synthesizer for generating safety-critical driving scenarios," 2019, *arXiv:1910.00099*.

[25] N. Ruiz, S. Schulter, and M. Chandraker, "Learning to simulate," 2018, *arXiv:1810.02513*.

[26] R. Lee, M. J. Kochenderfer, O. J. Mengshoel, G. P. Brat, and M. P. Owen, "Adaptive stress testing of airborne collision avoidance systems," in *Proc. IEEE/AIAA 34th Digit. Avionics Syst. Conf.*, 2015, pp. 6 C 2–1.

[27] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer, "Adaptive stress testing for autonomous vehicles," in *Proc. IEEE Intell. Veh. Symp.*, 2018, pp. 1–7.

[28] M. Klischat and M. Althoff, "Generating critical test scenarios for automated vehicles with evolutionary algorithms," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 2352–2358.

[29] B. Chen and L. Li, "Adversarial evaluation of autonomous vehicles in lane-change scenarios," 2020, *arXiv:2004.06531*.

[30] Y. Abeysirigoonawardena, F. Shkurti, and G. Dudek, "Generating adversarial driving scenarios in high-fidelity simulators," in *Proc. Int. Conf. Robot. Automat*, 2019, pp. 8271–8277.

[31] T. A. Wheeler and M. J. Kochenderfer, "Critical factor graph situation clusters for accelerated automotive safety validation," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 2133–2139.

[32] J. Norden, M. O'Kelly, and A. Sinha, "Efficient black-box assessment of autonomous vehicle safety," 2019, *arXiv:1912.03618*.

[33] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic, "Multilevel splitting for estimating rare event probabilities," *Operations Res.*, vol. 47, no. 4, pp. 585–600, 1999.

[34] P. Auer, "Using confidence bounds for exploitation-exploration tradeoffs," *J. Mach. Learn. Res.*, vol. 3, no. Nov, pp. 397–422, 2002.

[35] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 2249–2257.

[36] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017, pp. 16–17.

[37] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel, "Vime: Variational information maximizing exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1109–1117.

[38] D. Pathak, D. Gandhi, and A. Gupta, "Self-supervised exploration via disagreement," 2019, *arXiv:1906.04161*.

[39] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," 2016, *arXiv:1605.08803*.

[40] C. Winkler, D. Worrall, E. Hoogeboom, and M. Welling, "Learning likelihoods with conditional normalizing flows," 2019, *arXiv:1912.00042*.

[41] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," 2017, *arXiv:1711.03938*.

[42] W. G. Najm *et al.*, "Pre-crash scenario typology for crash avoidance research," united states. national highway traffic safety administration, Tech. Rep., 2007.

[43] "Carla Scenario Runner Library," https://github.com/carla-simulator/scenario_runner.

[44] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," 2019, *arXiv:1911.07602*.

[45] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," 2019, *arXiv:1910.01708*.

[46] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[47] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

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

[49] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[50] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018, *arXiv:1801.01290*.

[51] L. Kaiser *et al.*, "Model-based reinforcement learning for atari," 2019, *arXiv:1903.00374*.