How Do We Fail? Stress Testing Perception in Autonomous Vehicles

Harrison Delecki[1], Masha Itkina[1], Bernard Lange[1], Ransalu Senanayake[2], Mykel J. Kochenderfer[1]

Abstract— **Autonomous vehicles (AVs) rely on environment perception and behavior prediction to reason about agents in their surroundings. These perception systems must be robust to adverse weather such as rain, fog, and snow. However, validation of these systems is challenging due to their complexity and dependence on observation histories. This paper presents a method for characterizing failures of LiDAR-based perception systems for AVs in adverse weather conditions. We develop a methodology based in reinforcement learning to find likely failures in object tracking and trajectory prediction due to sequences of disturbances. We apply disturbances using a physics-based data augmentation technique for simulating LiDAR point clouds in adverse weather conditions. Experiments performed across a wide range of driving scenarios from a real-world driving dataset show that our proposed approach finds high likelihood failures with smaller input disturbances compared to baselines while remaining computationally tractable. Identified failures can inform future development of robust perception systems for AVs.**

I. INTRODUCTION

Autonomous vehicles (AVs) rely on perception systems to reason about critical information in their surroundings, such as the presence of other vehicles and their future behavior. These systems must perform reliably in a wide variety of real-world scenarios that may not be be present during development or may occur infrequently. Reliable perception performance in adverse weather continues to be a challenge for state-of-the-art perception systems [1], [2]. Identifying how and when these systems fail in such conditions is critical to the development and eventual deployment of autonomous systems in human environments.

Many state-of-the-art AV perception systems rely on independent modules to perform object detection, tracking, and trajectory prediction [3]–[5]. These modules may use data-driven or classical methods. Verifying the input-output properties of an individual module is not sufficient to guarantee good performance, since small errors in tasks like object detection may be magnified in downstream modules like trajectory prediction. Additionally, these systems rely on sequential observations of the environment, meaning that failures are associated with trajectories of detected objects. Searching for failures over trajectories is difficult due to the high-dimensionality of the search space.
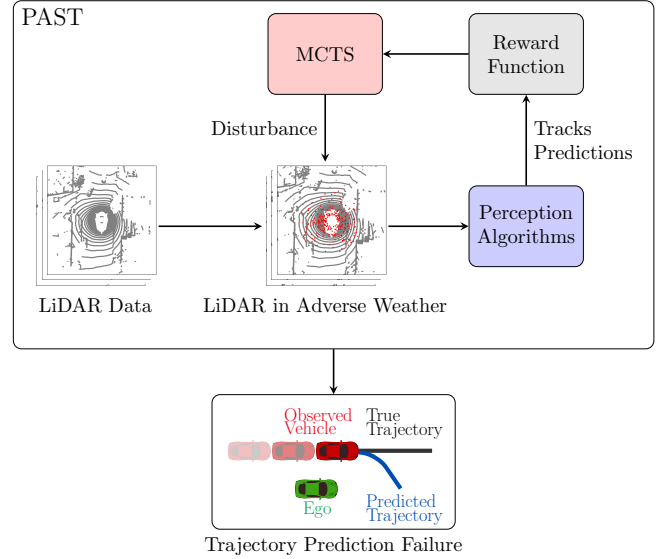
Fig. 1: Our proposed *PAST* framework for validation of tracking and prediction stages of an AV perception systems under adverse weather conditions. Red points indicate disturbances to the perception system caused by LiDAR beams reflecting off of airborne particles. MCTS controls how sequences of disturbances are added to the system to cause failures in perception tasks that depend on detections aggregated over time, such as object tracking and trajectory prediction. PAST seeks to find the most likely failures in these sequential perception system tasks.

Many previous approaches to validation of perception systems draw on the idea of an adversarial attack on object detection [6]–[8]. These methods solve an optimization problem to find a small perturbation or physical object that leads to an incorrect object detection when added to the base input. However, they do not consider the impact that such perturbations could have on downstream perception tasks that depend on a history of detections, such as object tracking and prediction. There have been several studies on the impact of weather disturbances on sensing and object detection [9], [10]. The performance of state-of-the-art object detectors can be severely degraded in the presence of rain, fog, or snow [1]. These studies primarily consider the robustness of sensing capabilities and object detection, but not the potential impact of weather on downstream perception tasks.

This work focuses on the validation of LiDAR-based perception systems in adverse weather. Research in validation of complex systems has led to the development of adaptive stress testing (AST), which uses reinforcement learning to find the most likely ways that a system can fail [11]. AST is used to validate black-box systems, where no knowledge of

the system under test is available. The reinforcement learning agent adds sequences of disturbances to the system to try to cause failure. However, existing work on AST has primarily considered stress testing of decision-making systems, where the goal is to drive the true state of the system to failure. In contrast, our goal is to drive a system's estimated state to a failure condition.

Our approach frames perception system validation as a sequential decision making problem. We extend the AST framework by using reinforcement learning to search for sequences of disturbances that lead to perception system failure. Our method is able to find sequences of disturbances introduced by models of adverse weather that cause the perception system to incorrectly track and predict the behavior of surrounding vehicles.

This paper makes the following contributions: We present a formulation of AST for perception systems which we call Perception AST (PAST), illustrated in Fig. 1. We apply PAST to find failures in a state-of-the-art AV LiDAR-based perception system under adverse weather conditions. Our experiments over many driving scenes show that our method can find more likely failures compared to other approaches.

II. RELATED WORK

A. Validation Methods

Safety validation is the process of ensuring safe operation of a system in an operating environment [12]. Methods such as statistical model checking have been used to estimate the probability that a perception system's estimates comply with some specifications [13]. Recently, AST has been applied to autonomous systems to search for the most likely ways that decision-making systems can fail [11], [14], [15]. This work extends the AST formulation to search for the most likely failures in perception systems rather than planning and control systems.

B. Adversarial Attacks

There has been a significant amount of work investigating robustness of image-based object detection with deep neural networks [6], [16]. These methods typically introduce local or global perturbations on inputs that cause networks to incorrectly classify or miss objects that would otherwise be detected. For more on these methods, refer to the survey by Akhtar *et al.* [17]. Generative adversarial networks (GANs) have also been used to generate attacks on object detection and to identify out-of-distribution examples in object detection [18], [19]. These methods require ample data to generate realistic attacks which may not be available and do not consider temporal sequences of observations.

Adversarial attack methods for image-based detectors have inspired methods for point-cloud based representations [20]. Point-cloud adversarial attacks can use similar perturbations as images, such as adding noise to point positions or completely removing some points from the input [7], [21]. These adversarial methods can quantify the robustness of deep-learning based object detectors, but do not consider the

impact of disturbances on downstream temporal tasks such as object tracking and prediction.

C. Impact of Weather on Perception

Prior works studied the impact of adverse weather such as rain, fog, and snow on LiDAR data [2], [9], [10]. Adverse weather tends to decrease the maximum detection range, add noise to range measurements, and introduce backscattering where LiDAR beams reflect off of particles in the air. Further studies have investigated building and applying models of adverse weather conditions to evaluate the robustness of object detection [1], [22], [23]. We draw on these models to simulate the impact of adverse weather conditions on LiDAR data. Specifically, these weather models provide controllable disturbances to perception system input data that AST uses to induce failures.

III. METHOD

This section describes our Perception Adaptive Stress Testing (PAST) formulation. First, we overview the typical components of an AV perception system. Next, we define the objective function for PAST, and show how it can be framed as a Markov decision process (MDP).

A. Problem Setting

This work considers LiDAR-based AV perception systems with modular components for object detection, tracking, and prediction. At each time step, the perception system takes as input a 3D LiDAR point cloud and produces a list of object tracks and motion predictions. We assume that object tracks are represented by 3D bounding boxes with associated position, velocity, and orientation. Predictions are made for all tracked vehicles that are not parked. Trajectory predictions for individual objects are represented by a time series of future positions. Our goal is to find likely sequences of disturbances to the input LiDAR data that cause large errors, or failures, in the object tracks and predictions.

B. Adaptive Stress Testing for Perception Systems

AST is a configuration of model-free reinforcement learning for black-box system validation [11]. Rather than learning a policy that optimizes an agent's performance, AST optimizes disturbances to the environment that cause an agent to fail. Previous applications of AST have focused on validation of decision-making systems, where the goal is to drive the true state of a system to failure. We present a formulation of AST for the validation of perception systems called Perception AST (PAST). In contrast to AST, the goal of PAST is to drive the estimated state of a system to failure.

We define a generative black box simulator $\mathscr{S}$ comprised of time-series sensor data, a stochastic disturbance model, and a perception system. We denote the hidden internal state of the system at time t as s_t. The internal state represents the perception system's belief or state estimate about the true states and future trajectories of surrounding agents. The simulation is stepped through time by drawing a random next state, where the sampling is pseudorandomly generated

from a provided seed. In our simulator, we first sample an input disturbance from the disturbance model and then use the perturbed data to update the perception system's state estimate. The simulator exposes the following four functions for simulation control:

- *Initialize($\mathscr{S}$)*: Resets the simulator $\mathscr{S}$ to an initial state s_0. This function resets the internal state of the perception system and the sensor data.
- *Step($\mathscr{S}, a$)*: Steps the simulation $\mathscr{S}$ in time by updating the perception system with a sample from the disturbance model. The randomness in the disturbance is controlled by the psuedorandom seed a. This function returns the likelihood of the transition, which is the likelihood of the sampled disturbance.
- *IsTerminal($\mathscr{S}$)*: Returns true if simulator $\mathscr{S}$ has reached the end of the simulation horizon and false otherwise.
- *IsFailure($\mathscr{S}$)*: Returns true if the perception system in simulator $\mathscr{S}$ has reached a failure.

The goal of PAST is to find the most likely sequence of disturbances generated by seeds $a_{0:t}$ such that *IsFailure($\mathscr{S}$)* is true. Equivalently, we can formulate the objective as finding the sequence of pseudorandom seeds that maximizes the joint probability of disturbances subject to the constraint that the disturbances lead to a failure.

Following the AST framework, we recast this problem into an MDP. An MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, R, T)$ [24]. An agent chooses an action $a \in \mathcal{A}$ based on a state $s \in \mathcal{S}$ and receives a reward r based on the reward function $R(s, a)$. The state transitions stochastically to the next state s' according to the transition model $T(s' \mid s, a)$. In the PAST MDP, actions correspond to psuedorandom seeds for the generation of disturbances to the perception input data. Recall that since the actions a control the stochastic transition of the system, the sequence of actions uniquely determines the system's state s. The transition model for the MDP is represented by the *Step* function exposed by the simulator, which takes in an action and returns the probability of the transition.

The reward function is designed to be equivalent to maximizing the joint probability of all actions, assuming that each action is independent. The functions *IsFailure* and *IsTerminal* exposed by the simulator are used to calculate the reward according to:

$$R(s, a) = \begin{cases} 0 & \text{if } \textit{IsFailure}(s) \\ \log p(a) & \text{else if not } \textit{IsTerminal}(s) \\ -\alpha & \text{otherwise} \end{cases} \quad (1)$$

where α is a large term that penalizes cases where a terminal state is reached that is not a failure. Assuming actions are independent, the agent in the MDP is maximizing the sum over $\log p(a_t)$, which is equivalent to maximizing the product over $p(a_t)$ or the probability of the action sequence.

The goal of a PAST agent is to maximize its expected utility by finding a policy π that specifies an action $a = \pi(s)$. The utility of following a policy π from state s is given by

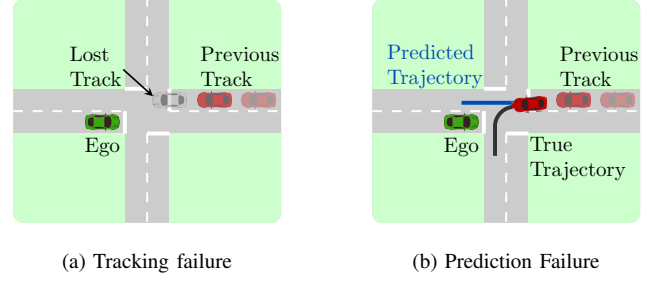
(a) Tracking failure   (b) Prediction Failure

Fig. 2: Illustration of tracking failure (left) and prediction failure (right). In both images, the ego vehicle is in blue and the observed vehicle is in red.

the value function:

$$V^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s'} T(s' \mid s, \pi(s)) V^\pi(s') \quad (2)$$

where γ is the discount factor that controls the weight of future rewards. Algorithms such as Monte Carlo Tree Search (MCTS) can be used to find an optimal policy. MCTS is an online sampling-based algorithm that can be used to find solutions to MDPs [25]. MCTS builds a search tree by sampling the state and action spaces, and estimates the value of states and actions through forward simulation. This work uses a variant of MCTS with double progressive widening (DPW) [26]. DPW regulates the branching factor in the state and action spaces to prevent the number of children in the search tree from exploding when the state or action spaces are very large.

IV. EXPERIMENTS

In this section we present the experimental setup used to illustrate the performance of our proposed method. We describe specific data sources, perception system components, and failure definitions for each experiment. We consider two experimental settings. We first illustrate the effectiveness of our approach at finding failures in the perception of a single vehicle. Then we consider an at scale experiment using an adverse weather disturbance model.

A. Experimental Setup

We perform validation of an AV perception system using data from the real-world driving dataset, nuScenes [27]. The dataset contains many hours of real-world driving data divided into $20\,\mathrm{s}$ long scenes. Each scene contains 32-beam LiDAR sweeps at $20\,\mathrm{Hz}$ and ground truth 3D object bounding boxes. We only consider the 123 scenes from the validation split that were recorded in clear weather conditions. We use the validation split to avoid any issues that may be caused by overfitting in perception modules. Our algorithm does not involve training or tuning hyperparameters based on this data. We treat each of these scenes as an independent validation case, in which we stress test the system to find a likely sequence of disturbances that lead to failure. Each scene is simulated by stepping through time, perturbing the recorded point cloud, and updating the perception system.
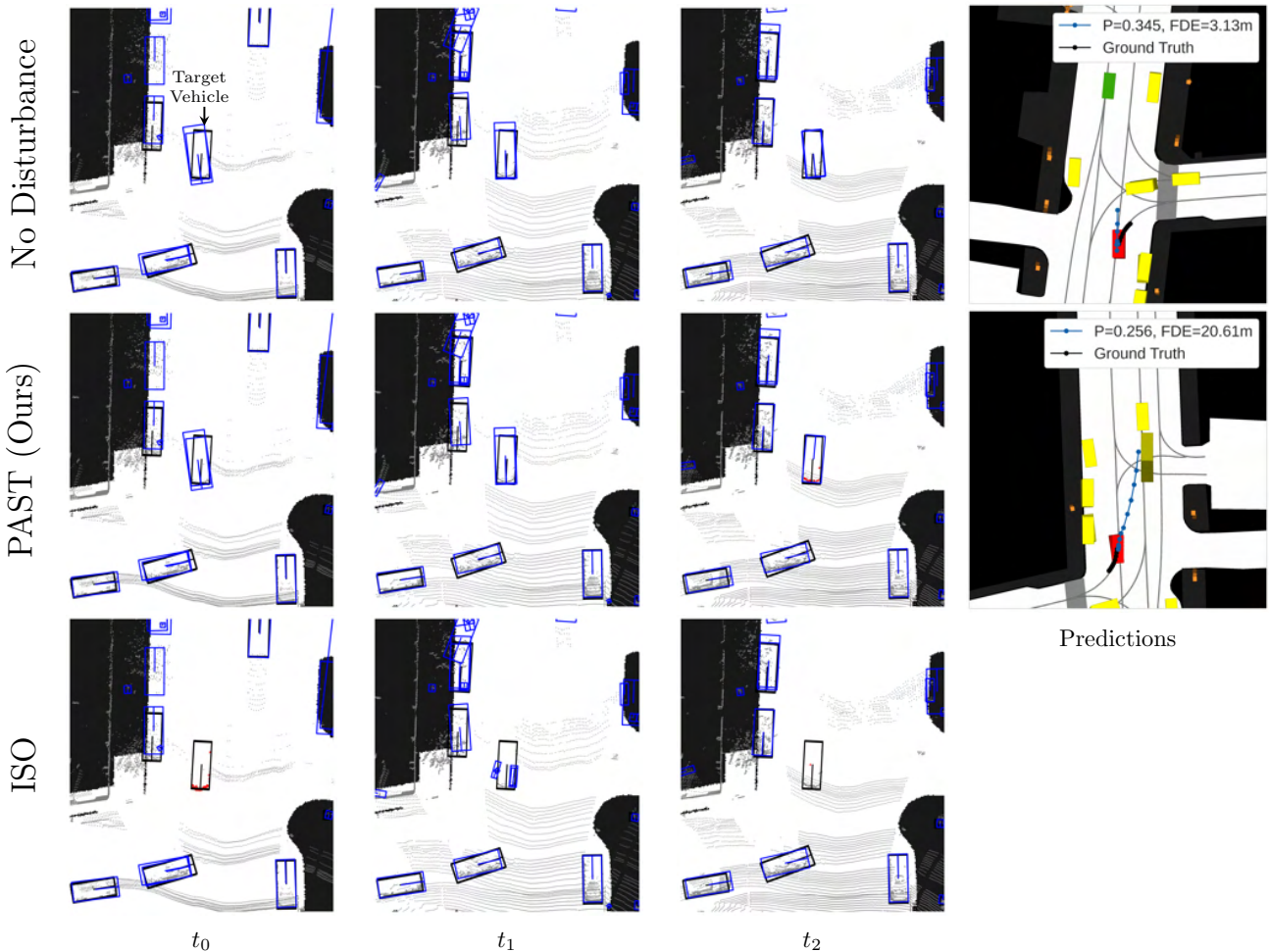
Fig. 3: Detections (blue) and ground truth (black) bounding boxes leading up to the failure discovered in the single object perception PAST experiment. The target vehicle is in the center of each detection frame. The solid line inside the box indicates heading. PAST does not remove any points until t_3, where it removes 15 points from the front of the target vehicle. This disturbance causes a large error (almost 180°) in the detected heading of the target vehicle. Under this simple disturbance identified by PAST, the target vehicle is predicted to be travelling in the opposite direction as the ground truth. The ISO baseline removes points at each time step that prevent the target from being detected. Since there are no detections of the target vehicle with ISO, the perception system can't make a trajectory prediction. The target vehicle is shown in red in the predictions. AST is able to find a more likely path to failure by considering disturbance trajectories.

B. Perception System Under Test

We consider validation of state-of-the-art detection, tracking, and prediction modules that are commonly used to baseline nuScenes perception tasks. Object detection is performed by the PointPillars architecture [28], [29]. Multi-object tracking is performed by AB3DMOT, which uses a Kalman filter to maintain tracks of 3D bounding box observations [30]. Finally, trajectory prediction is performed by CoverNet [31], which makes predictions based on a pre-determined set of candidate trajectories.

C. Failure in Single Object Perception

As an illustrative example, we consider the case where disturbances only interfere with the perception of a particular vehicle in the scene, which we refer to as the "target vehicle". We hand select a target vehicle from the dataset, and define a simplified disturbance model that only impacts points inside the target vehicle's bounding box.

Disturbance Model: We assume that every LiDAR point in the bounding box is associated with an independent Bernoulli random variable representing whether the point will be removed. All points have the same probability of being removed, θ. Given a random sample generated with seed a for all m available points, the likelihood of the disturbance is:

$$p(a) = \theta^n (1-\theta)^{m-n} \qquad (3)$$

where n is the number of points selected to be removed. For demonstration purposes, we use $\theta = 0.1$.

Failure Definition: We define a failure in perception for this simple example in terms of the target vehicle's estimated track and trajectory prediction. These definitions are illustrated in Fig. 2. We define failures in tracking to occur when the error in the estimated position exceeds $2\,\mathrm{m}$, or when the track associated with the target vehicle is lost. We define a failure in prediction to occur when the final displacement

TABLE I: PAST and baseline results on the nuScenes dataset. Mean and standard error values are provided for the local disturbance magnitude δ, global disturbance magnitude Δ, and trajectory length. Higher failure rate indicates better performance, while lower values of disturbance magnitudes and trajectory length are desired. PAST is able to find many likely failures in both tracking and prediction.

Failure Criteria	Method	Failure Rate (%)	δ (%)	Δ (%)	Trajectory Length (-)
Tracking and Prediction	ISO (Baseline)	69.1	8.1 ± 2.1	0.11 ± 0.20	5.75 ± 0.57
	MC (Large)	25.1	$4.68 \times 10^{-2} \pm 7.6 \times 10^{-3}$	3.94 ± 0.35	12.2 ± 1.2
	MC (Small)	22.1	$3.08 \times 10^{-2} \pm 4.5 \times 10^{-3}$	3.54 ± 0.54	13.9 ± 1.3
	PAST (Large)	51.2	$4.06 \times 10^{-2} \pm 1.2 \times 10^{-2}$	4.10 ± 0.42	7.54 ± 0.62
	PAST (Small)	31.7	$2.71 \times 10^{-2} \pm 8.2 \times 10^{-3}$	4.00 ± 0.53	8.63 ± 0.57
Prediction	MC (Prediction)	21.9	$5.18 \times 10^{-2} \pm 7.1 \times 10^{-3}$	3.19 ± 0.57	16.5 ± 1.1
	PAST (Prediction)	29.3	$7.70 \times 10^{-2} \pm 1.5 \times 10^{-2}$	3.41 ± 0.55	9.46 ± 0.71

error exceeds $15\,\mathrm{m}$ in CoverNet's most likely predicted trajectory. We chose this threshold to capture cases where the predicted intent of a vehicle is significantly different from ground truth. In practice, these failure thresholds should be decided by vehicle manufacturers and policy makers.

D. At Scale Validation in Adverse Weather

In the following set of experiments, we perform validation on all 123 scenes from the nuScenes dataset with disturbances based on an adverse weather model. These experiments demonstrate the ability of our method to scale validation of a state-of-the-art perception system over a wide range of driving scenes.

Disturbance Model: We use the LiDAR Light Scattering Augmentation (LISA) software package to model the effects of adverse weather on LiDAR data [23]. LISA provides methods to augment LiDAR measurements with physics-based models of rain, fog, and snow. In our experiments, we focus on disturbances due to rain. LISA takes as input a rain rate and returns a new point cloud with simulated rain effects. The algorithm generates a new point cloud by randomly sampling from physics-based distributions to remove points, add range noise, and reflect points. For use within our PAST framework, we modify LISA to accept a random seed and to compute the log-likelihood of sampled disturbances.

Failure Definition: The definitions of perception failure are slightly different in adverse weather experiments to more accurately reflect how a real-world perception system might be used. Here, failures in prediction occur when the minimum final displacement error exceeds $15\,\mathrm{m}$ over Cover-Net's top five most confident predictions. The definition of tracking failures is the same as in the single object perception experiment. We only check for failures that emerge due to the adverse weather disturbances. In particular, if a failure criterion is met for a vehicle without disturbances, we do not terminate PAST for this failure event.

We perform PAST over all 123 scenes using three different configurations. The first configuration we call PAST (Large). PAST selects from three relatively heavy rain rates of 20, 30, and $40\,\mathrm{mm/h}$. We consider more mild disturbances in PAST (Small), where we consider rain rates of 5, 10, and $15\,\mathrm{mm/h}$. The last experiment PAST (Prediction) uses the same smaller rain rates, but only considers failures in prediction. PAST

(Prediction) demonstrates the flexibility of PAST to find failures under different criteria. For all PAST experiments, we use MCTS with a maximum of $2,000$ iterations.

E. Baselines

As a baseline for our approach, we consider a modified version of the Iterative Salience Occlusion (ISO) algorithm, an adversarial attack on 3D object detection [21]. ISO uses latent feature spaces of 3D object detectors to identify sets of critical points, or points in the input space that contribute the most to the network's identification of an object. ISO iteratively removes these critical points from the input until the network misclassifies the result. The original ISO algorithm was created for single object detection. We use a modified version of the ISO algorithm for the multi-object detection task in AV perception. During simulation, we run ISO each time step until a maximum number of iterations is reached or a vehicle is misclassified. Rather than checking for misclassification of a single object, we check for misclassification of any agent of a specific class, such as 'car'. After termination of ISO, we use the new point cloud that it returns to update the perception system's state estimate. Adversarial attack methods like ISO struggle to stay computationally tractable when considering multiple time steps and very large point clouds. We restrict ISO to consider LiDAR points inside the ground truth bounding boxes of other vehicles and set a limit of 100 iterations to ensure computational tractability in our experiments.

We also baseline our approach using Monte Carlo (MC) random search with the adverse weather disturbance model to confirm that PAST is able to maximize its objective. Random search selects actions at random using the same number of iterations as MCTS and returns the maximum likelihood failure discovered. This baseline is repeated for each of the large rain rate, small rain rate, and prediction-only experiment cases that we consider for PAST.

V. RESULTS

Experiments were conducted on a desktop with 32GB of RAM, an Intel i7-7700K CPU, and an NVIDIA GeForce GTX 1080 Ti GPU. PAST was implemented using AST-Toolbox, an open-source software package for designing
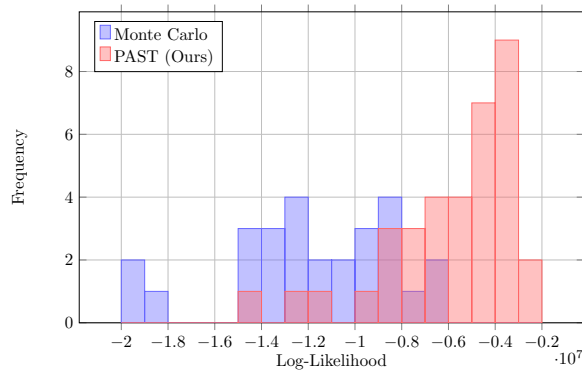
Fig. 4: Histogram of the total log-likelihood of failures in trajectory prediction found using Monte Carlo random search and PAST. PAST successfully finds more likely failures than random search.

and running AST experiments.[1] We first consider an experiment involving perception of a single object in a scene. Next, we perform experiments over many driving scenes using an adverse weather disturbance model, demonstrating our method's ability to scale. Our code is available at: https://github.com/sisl/PerceptionAdaptiveStressTesting.

Failure in Single Object Perception: The object detections and predictions for the single object perception experiment are illustrated in Fig. 3. In the detections, the target vehicle is shown in the center of the frame. The target vehicle is making a left turn while becoming occluded with respect to the ego vehicle. In the predictions, the target vehicle is shown in red. The perception system successfully outputs good tracks and predictions without disturbances as seen in the first row.

The most likely failure event according to PAST is shown in the second row in Fig. 3. The most likely failure in this experiment corresponds to the failure that results from the fewest total number of points removed. At the beginning of its search, PAST removes points at each time step. Over successive iterations, PAST seeks to maximize the expected reward, or minimize the number of points removed. PAST discovers that a failure can occur by only removing 15 points at the third time step (as shown in Fig. 3). The ISO baseline removes 112 LiDAR points in total, removing points at every time step, resulting in the target vehicle never being detected by definition of the approach. By reasoning over trajectories of disturbances, PAST is able to find a failure trajectory that removes significantly fewer points than the ISO baseline.

At Scale Validation in Adverse Weather: For the at scale experiment, we compare the performance of our method against baselines in terms of failure rate, mean local disturbance magnitude δ, global disturbance magnitude Δ, and mean trajectory length. Failure rate refers to the proportion of scenes that the method was able to find failures in. Local disturbance magnitude δ is the proportion of points removed from inside the bounding box of the vehicle involved in the failure. Global disturbance magnitude Δ is the proportion

of points moved or removed in the whole point cloud. Trajectory length refers to the number of observations of the failure vehicle leading up to the failure event.

A summary of experimental results for PAST and baseline methods over the nuScenes validation split is shown in Table I. The first five rows in Table I show results for experiments considering failures in tracking and prediction. PAST outperforms MC in failure rate as well as in trajectory length. The baseline ISO algorithm is substantially more aggressive in the number of points removed locally than our algorithm, resulting in a higher failure failure rate when considering failures in tracking and prediction. For ISO to be tractable at the scale of these experiments, it was restricted to consider points inside ground truth bounding boxes. ISO tends to remove a higher proportion of LiDAR points associated with specific vehicles to introduce poor detections. In contrast, while PAST adds a disturbance over the whole point cloud, this disturbance is relatively small with respect to the ground truth vehicles as demonstrated by the δ metric. Therefore, PAST can scale validation to larger datasets and add smaller disturbances by considering longer failure trajectories.

When considering failures in tracking and prediction, all of the failures found by PAST and the baseline methods occur in object tracking. To find failures in prediction, there must first be tracks to predict. Since the disturbances cause poor detections, it is more difficult to maintain good object tracks and the perception system fails in tracking before failures in prediction can manifest. The last two rows of Table I show results considering only failures in prediction. The ISO algorithm was unable to find failures in prediction. Both MC (Prediction) and PAST (Prediction) were successful, since they are able to consider longer failure trajectories. PAST finds more failures in predictions than MC because it is able learn sequences of actions that maximize the PAST objective function. Fig. 4 shows a histogram comparing the likelihood of failures found in MC (Prediction) and PAST (Prediction). In addition to being able to find more failures than MC, PAST also finds failures that are shorter and more likely. The high performance of PAST compared to random search suggests that PAST is maximizing its objective function.

An example failure trajectory is illustrated in Fig. 5. In this scene, the ego vehicle is stopped at a light with a few surrounding vehicles. The vehicle just to the left of the ego begins to accelerate forward. Without disturbances, this behavior is predicted accurately. However, the rain disturbance identified by our PAST method makes it appear to the perception system that the vehicle is turning right and into the ego vehicle's lane, cutting the ego vehicle off. This incorrect prediction about the other vehicle's future trajectory could lead to undesirable performance in the AV, such as an unnecessary hard braking maneuver.

PAST required $90\,\mathrm{min}$ for a single scene on average, while Monte Carlo and ISO averaged $60\,\mathrm{min}$ and $200\,\mathrm{min}$, respectively. While ISO finds shorter paths to failure than MC or PAST, its aggressive local disturbances are not likely to occur in the real world. Considering smaller disturbances
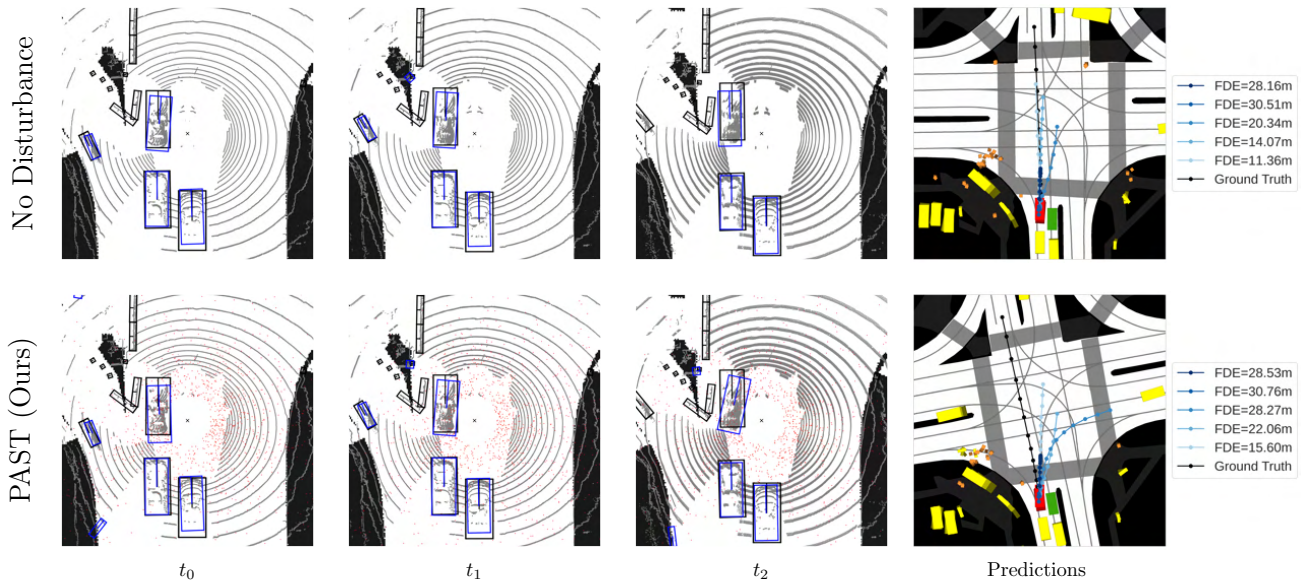
Fig. 5: Comparison of detections and predictions without disturbances (top) and with disturbances found by PAST (bottom). Red points indicate points reflected due to the adverse weather model. Detected bounding boxes are shown in blue and the ground truth is shown in black. With the adverse weather disturbance, the detections of the vehicle just to the left of the ego vehicle appear to be turning right. Predictions are shown in the last column with the ego vehicle in green and the predicted vehicle in red. The observations under disturbances cause the predicted trajectory to be turning right and in front of the ego vehicle when the vehicle is actually accelerating straight ahead. This type of failure could cause the AV to brake unnecessarily in traffic, causing a potentially unsafe driving scenario. Results for ISO are not shown since it is unable to find a failure.

with ISO becomes computationally intractable due to the high dimensionality of the search space. PAST is able to find failures in prediction and tracking that are relatively small while remaining computationally tractable.

VI. CONCLUSION

AVs rely on perception and prediction to reason about their surroundings. Identifying how and when these systems fail in adverse weather conditions is critical to the development and deployment of autonomous systems in human environments. This work presents PAST, a method for validation of LiDAR-based perception systems that uses reinforcement learning to find likely failures. The method was applied to the validation of a perception system in adverse weather using real-world LiDAR data. The results showed that the proposed PAST method tractably finds likely disturbances that introduce large errors into the tracking and prediction of other vehicles across a range of driving scenarios. A key future research direction is to apply PAST to validate end-to-end perception and prediction methods [32]–[35], enabling quantitative comparison of the robustness between modular and end-to-end perception systems under adverse weather conditions. Additionally, understanding failures in different systems could inform how to combine perception systems to balance each component's strengths and weaknesses, resulting in a more robust solution.

REFERENCES

[1] M. J. Mirza, C. Buerkle, J. Jarquin, M. Opitz, F. Oboril, K.-U. Scholl, and H. Bischof, "Robustness of object detectors in degrading weather conditions," in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2021, pp. 2719–2724.

[2] S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, and M. A. Kaafar, "The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car," *IEEE Vehicular Technology Magazine*, vol. 14, no. 2, pp. 103–111, 2019.

[3] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3D object detection methods for autonomous driving applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, 2019.

[4] J. Van Brummelen, M. O'Brien, D. Gruyer, and H. Najjaran, "Autonomous vehicle perception: The technology of today and tomorrow," *Transportation Research Part C: Emerging Technologies*, vol. 89, pp. 384–406, 2018.

[5] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 33–47, 2022.

[6] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *International Conference on Computer Vision (ICCV)*, IEEE, 2017, pp. 1378–1387.

[7] C. Xiang, C. R. Qi, and B. Li, "Generating 3D adversarial point clouds," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 9136–9144.

[8] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, F. Tramèr, A. Prakash, T. Kohno, and D. Song, "Physical adversarial examples for object detectors," in *USENIX Conference on Offensive Technologies*, 2018, p. 1.

[9] M. Kutila, P. Pyykönen, H. Holzhüter, M. Colomb, and P. Duthon, "Automotive LiDAR performance verification in

fog and rain," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 1695–1701.

[10] Y. Li and J. Ibanez-Guzman, "LiDAR for autonomous driving: The principles, challenges, and trends for automotive LiDAR and perception systems," *IEEE Signal Processing Magazine*, vol. 37, no. 4, pp. 50–61, 2020.

[11] R. Lee, M. J. Kochenderfer, O. J. Mengshoel, G. P. Brat, and M. P. Owen, "Adaptive stress testing of airborne collision avoidance systems," in *Digital Avionics Systems Conference (DASC)*, 2015.

[12] A. Corso, R. Moss, M. Koren, R. Lee, and M. Kochenderfer, "A survey of algorithms for black-box safety validation of cyber-physical systems," *Journal of Artificial Intelligence Research*, vol. 72, pp. 377–428, 2021.

[13] M. Barbier, A. Renzaglia, J. Quilbeuf, L. Rummelhard, A. Paigwar, C. Laugier, A. Legay, J. Ibañez-Guzmán, and O. Simonin, "Validation of Perception and Decision-Making Systems for Autonomous Driving via Statistical Model Checking," in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 252–259.

[14] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer, "Adaptive stress testing for autonomous vehicles," in *Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1–7.

[15] K. D. Julian, R. Lee, and M. J. Kochenderfer, "Validation of image-based neural network controllers through adaptive stress testing," in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2020, pp. 1–7.

[16] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[17] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14 410–14 430, 2018.

[18] X. Wei, S. Liang, N. Chen, and X. Cao, "Transferable adversarial attacks for image and video object detection," in *International Joint Conference on Artificial Intelligence (IJCAI)*, Jul. 2019, pp. 954–960.

[19] J. Nitsch, M. Itkina, R. Senanayake, J. Nieto, M. Schmidt, R. Siegwart, M. J. Kochenderfer, and C. Cadena, "Out-of-distribution detection for automotive perception," in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2021, pp. 2938–2943.

[20] D. Liu, R. Yu, and H. Su, "Extending adversarial attacks and defenses to deep 3D point cloud classifiers," in *International Conference on Image Processing (ICIP)*, IEEE, 2019, pp. 2279–2283.

[21] M. Wicker and M. Kwiatkowska, "Robustness of 3D deep learning in an adversarial setting," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 11 767–11 775.

[22] M. Bijelic, T. Gruber, F. Mannan, F. Kraus, W. Ritter, K. Dietmayer, and F. Heide, "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 11 679–11 689.

[23] V. Kilic, D. Hegde, V. Sindagi, A. B. Cooper, M. A. Foster, and V. M. Patel, "LiDAR light scattering augmentation (LISA): Physics-based simulation of adverse weather conditions for 3D object detection," *ArXiv*, 2021.

[24] M. J. Kochenderfer, T. A. Wheeler, and K. H. Wray, *Algorithms for Decision Making*. MIT Press, 2022.

[25] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

[26] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, and N. Bonnard, "Continuous upper confidence trees," in *Learning and Intelligent Optimization (LION)*, Springer Berlin Heidelberg, 2011, pp. 433–445.

[27] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 11 618–11 628.

[28] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019.

[29] OpenPCDet Development Team, *OpenPCDet: An open-source toolbox for 3D object detection from point clouds*, 2020.

[30] X. Weng, J. Wang, D. Held, and K. Kitani, "3D multi-object tracking: A baseline and new evaluation metrics," in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020.

[31] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal behavior prediction using trajectory sets," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020.

[32] M. Itkina, K. Driggs-Campbell, and M. J. Kochenderfer, "Dynamic environment prediction in urban scenes using recurrent representation learning," in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2019, pp. 2052–2059.

[33] M. Toyungyernsub, M. Itkina, R. Senanayake, and M. J. Kochenderfer, "Double-prong ConvLSTM for spatiotemporal occupancy prediction in dynamic environments," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 13 931–13 937.

[34] B. Lange, M. Itkina, and M. J. Kochenderfer, "Attention augmented ConvLSTM for environment prediction," in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2021, pp. 1346–1353.

[35] M. Liang, B. Yang, W. Zeng, Y. Chen, R. Hu, S. Casas, and R. Urtasun, "PnPNet: End-to-end perception and prediction with tracking in the loop," in *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020, pp. 11 550–11 559.