# A Appendix

## A.1 SafeBench statistics

We present the statistics of testing scenarios generated by each generation algorithm in Table 8. For each algorithm, we report the statistics both before and after scenario selection, where we only keep scenarios that have high transferability across AD algorithms. By applying the $4$ generation algorithms, we obtain $3,140$ testing scenarios in total, from which we select $2,352$ testing scenarios for AD evaluation.

## A.2 SafeBench design details

Our evaluation platform runs in the Docker container and is built upon the Carla simulator [28]. We design 4 components (nodes) that are highly flexible for users to customize: ego vehicle, agent node, scenario node, and evaluation node. These components communicate with each other through ROS. We detail the platform design as follows.

**Docker image** We provide a docker image containing SafeBench, which makes the platform more portable. The docker image is built based on Ubuntu 20.04. Inside the docker image, we have pre-installed Carla 0.9.11 and ROS Noetic for simulation and communication in SafeBench respectively.

**ROS services** The communication between different nodes is implemented using ROS services. For example, when the AD algorithm in the agent node is ready, it will request the waypoint information specified by the scenario node. The scenario node will send out the waypoint information in the current scenario once it receives a request from the agent node.

**CARLA** We use Carla 0.9.11 as our traffic simulator. The scenario runner [29] is incorporated in the scenario node to easily define and execute different scenarios. In the agent node, we develop our RL agent based on Gym Carla [89] environment which supports an OpenAI gym style interaction between the agent and Carla simulator.

## A.3 Definition of scenarios and examples of route variants

We first give detailed definitions of the $8$ traffic scenarios considered in SafeBench in Table 7 and screenshots of them in Figure 4. We also develop benign scenarios based on these safety-critical scenarios. In benign situations, everything is the same except that the other vehicles are auto-piloted. As a result, we have $8$ kinds of benign scenarios, and we can compare the benign performances with safety-critical ones.

We show more examples of route variants incorporated in our evaluation platform in Figures 5, 6 and 7.

## A.4 Evaluation metrics

We follow the equations introduced in Section 3.4 to calculate evaluation metrics. Specifically, for *route following stability*, we first set $x_{max}$ to $5$ and then calculate the expectation. For other metrics, we directly calculate the expectation of each variable over the scenario distribution $\mathcal{P}$. When calculating the *overall score*, we follow the maximum allowed value $m^i_{max}$ and weights $w^i$ for each metric $m^i$ given in Table 9. The weight for each metric depends on the evaluation level. Metrics in *Safety Level* are assigned the highest weights since they focus on serious violations of traffic rules. Among the $4$ safety level metrics, the weight of CR is $5$ times larger than others' weights. The weights of metrics in *Functionality Level* are one-half of the weights in *Safety Level*, while the weights in *Etiquette Level* are only one-fifth of them. Such a weight setup first emphasizes safety and then encourages the ego vehicle to complete the given tasks in a comfortable way.

## A.5 Implementation details of AD algorithms

**Reward function** During training, all RL algorithms share the same reward function. The reward is a weighted sum of 7 items. We set the weight of longitudinal speed to $1$, the weight of lateral

Table 7: Scenario Description

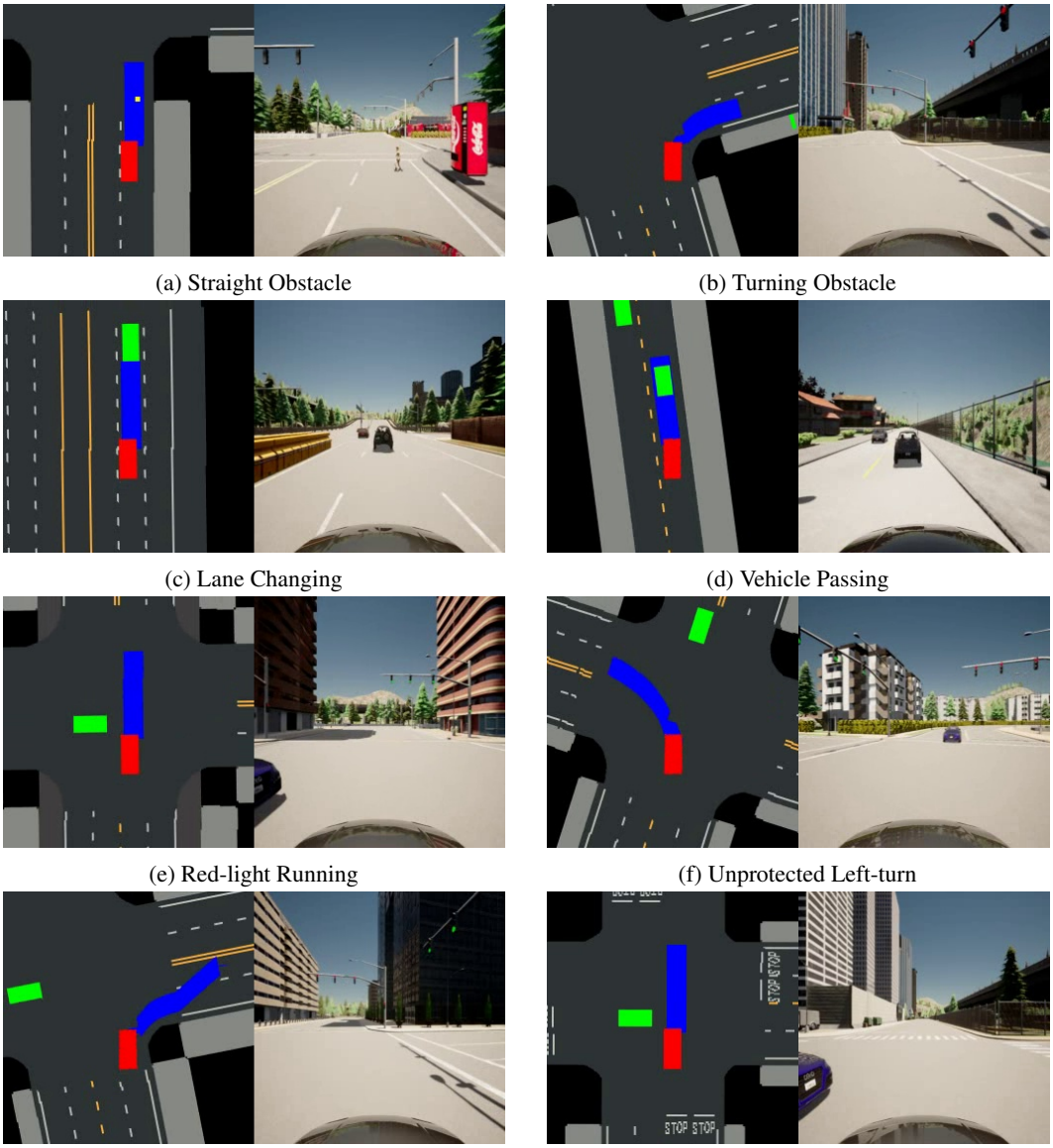
| **Scenario Name** | Description |
|---|---|
| Straight Obstacle | The ego vehicle encounters an unexpected cyclist or pedestrian on the road and must perform an emergency brake or an avoidance maneuver. As shown in Figure 4a, the vision of the ego vehicle is usually blocked by an obstacle, which is safety-critical since the reaction time left for the ego vehicle is very short. |
| Turning Obstacle | As shown in Figure 4b, while turning at an intersection, the ego vehicle finds an unexpected cyclist or pedestrian on the road and must perform an emergency brake or an avoidance maneuver. |
| Lane Changing | In this scenario, the ego vehicle should perform a lane changing to evade a leading vehicle, which is moving too slowly. In addition, there is another leading vehicle in the adjacent lane, which is traveling at a normal speed. The ego vehicle needs to avoid hitting both cars when overtaking. See Figure 4c for more details. |
| Vehicle Passing | The ego vehicle must go around a blocking object using the opposite lane, dealing with oncoming traffic. The ego vehicle should avoid colliding with both cars and also avoid driving outside the lane. We provide an example in Figure 4d. |
| Red-light Running | When the ego vehicle is going straight at an intersection, a crossing vehicle runs a red light. The ego vehicle is forced to take actions to avoid potential collisions as shown in Figure 4e. |
| Unprotected Left-turn | As shown in Figure 4f, the ego vehicle is performing an unprotected left turn at an intersection while there is a vehicle going straight in the opposite lane. |
| Right-turn | In this scenario, the ego vehicle is performing a right turn at an intersection, with a crossing vehicle in front. Collision avoidance actions must be taken to keep safe. We present an example in Figure 4g. |
| Crossing Negotiation | In this scenario, the ego vehicle meets another crossing vehicle when passing an intersection with no traffic lights. As shown in Figure 4h, the ego vehicle should negotiate with the other vehicle to cross the unsignalized intersection in an orderly and safe manner. |

Table 8: Statistics of SafeBench testing scenarios.

| **Algo.** | **Scenario Selection** | **Traffic Scenarios** | | | | | | | | **Total** |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Straight Obstacle | Left-turn Obstacle | Lane Changing | Vehicle Passing | Red-light Running | Unprotected Left-turn | Right-turn | Crossing Negotiation | |
| LC | Before | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 800 |
| | After | 41 | 13 | 100 | 99 | 42 | 69 | 59 | 58 | 481 |
| AS | Before | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 800 |
| | After | 68 | 42 | 100 | 100 | 72 | 86 | 53 | 64 | 585 |
| CS | Before | 100 | 100 | 90 | 90 | 90 | 90 | 90 | 90 | 740 |
| | After | 60 | 76 | 90 | 90 | 74 | 77 | 83 | 79 | 629 |
| AT | Before | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 800 |
| | After | 59 | 33 | 99 | 100 | 100 | 87 | 89 | 90 | 657 |
| Total | Before | 400 | 400 | 390 | 390 | 390 | 390 | 390 | 390 | 3140 |
| | After | 228 | 164 | 389 | 389 | 288 | 319 | 284 | 291 | 2352 |

acceleration to $0.2$, and the weight of steering to $5$. If the ego vehicle encounters a collision or drives out of lane, we give a reward of $-1$ as a penalty. If the speed of the ego vehicle is larger than a threshold, we give a reward of $-10$ as a penalty. The speed threshold is set to $9$. We also add a constant reward of $0.1$.

**Action space** Similarly, the action space of every RL model is the same, which includes acceleration and a steering value. For acceleration, the maximum and minimum allowed values are $3$ and $-3$, respectively. We limit the absolute value of steering to no greater than $0.3$. After having the acceleration and steering, we need to convert these values into Carla's vehicle control format, where



(a) Straight Obstacle (b) Turning Obstacle

(c) Lane Changing (d) Vehicle Passing

(e) Red-light Running (f) Unprotected Left-turn

(g) Right-turn (h) Crossing Negotiation

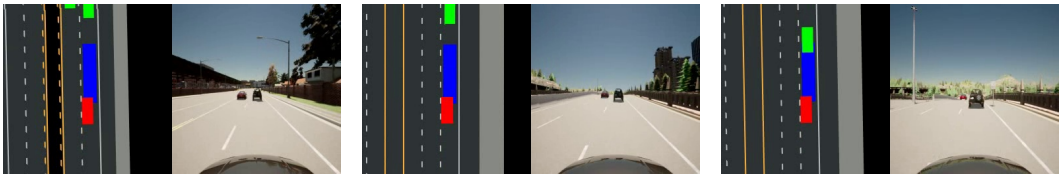
Figure 4: Pre-crash scenarios.

we need to calculate the throttle and brake of the ego vehicle. The throttle and brake are calculated using the following equations:

$$throttle = \begin{cases} acc/3, & acc > 0 \\ 0, & otherwise \end{cases}, brake = \begin{cases} 0, & acc > 0 \\ -acc/8, & otherwise \end{cases} \tag{2}$$

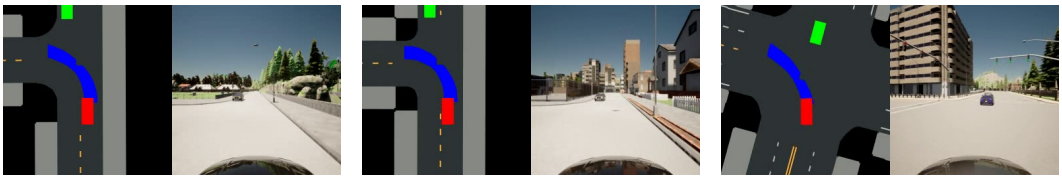
where $acc$ denotes the acceleration given by RL models. Both throttle and brake will be clipped to the interval $[0, 1]$.

**Model Architecture** The model we used for deep RL methods is a simple multi-layer perceptron. The size of the hidden layer is [256, 256]. When adding bird-eye view images or camera images into input information, we use a separate image encoder to extract image features. The encoder is end-to-end trained with the actor network in RL models. We provide more details about the architecture of the image encoder in Table 10.



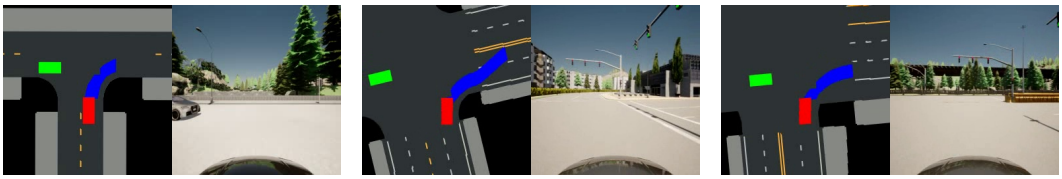
(a) Two-lane highway (b) Three-lane bridge (c) Three-lane bridge with a speed limit sign

Figure 5: Example route variants of scenario 3.



(a) Single-lane T-junction without surrounding buildings (b) Single-lane T-junction with surrounding buildings (c) Two-lane intersection

Figure 6: Example route variants of scenario 6.



(a) Single-lane T-junction (b) Two-lane intersection (c) Two-lane T-junction

Figure 7: Example route variants of scenario 7.

**DDPG hyperparameters** The policy learning rate is 0.0003, and the Q-value learning rate is 0.001. The standard deviation for Gaussian exploration noise added to the policy at training time is 0.1. The discount factor is 0.99. The number of models in the Q-ensemble critic is 1.

**SAC hyperparameters** The policy learning rate and Q-value learning rate are set to be 0.001. The entropy regularization coefficient, which is equivalent to the inverse of the reward scale in the original SAC paper, is 0.1. The discount factor equals 0.99, and the number of models in the Q-ensemble critic is 2.

**TD3 hyperparameters** The policy learning rate and Q-value learning rate are set to 0.001. The standard deviation for Gaussian exploration noise added to the policy at training time is 0.1. The standard deviation for smoothing noise added to noise is 0.2 The limit for the absolute value of smoothing noise is 0.5. Policy update delay is 2. The discount factor is 0.99. The number of models in the Q-ensemble critic is 2.

**PPO hyperparameters** The policy learning rate is 0.0003, and the Q-value learning rate is 0.001. The clip ratio of the policy object is 0.2. The target KL divergence is 0.01. We set both actor and critic training iterations to 80. The discount factor is 0.99, and the number of interaction steps is 1000.

### A.6 Training details of AD algorithms

All of the 4 deep RL algorithms are trained in Carla town03. Because town03 is the most complex town, with a 5-lane junction, a roundabout, unevenness, a tunnel, and more, according to Carla's official document. The number of warm-up steps for off-policy methods is 600. The interpolation factor in polyak averaging for the target network is 0.995. The number of training epochs is different

Table 9: Constants and weights used in SafeBench evaluation metrics.

| Symbol | Safety Level | | | | Functionality Level | | | Etiquette Level | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | CR | RR | SS | OR | RF | Comp | TS | ACC | YV | LI |
| $m^i_{max}$ | 1 | 1 | 1 | 50 | 1 | 1 | 60 | 8 | 3 | 20 |
| $w^i$ | 0.495 | 0.099 | 0.099 | 0.099 | 0.050 | 0.050 | 0.050 | 0.020 | 0.020 | 0.020 |

Table 10: Model architecture of image encoder.

| Layer | Input Channels | Output Channels | Kernel Size | Stride | Padding |
|---|---|---|---|---|---|
| Convolution Layer 1 | 3 | 32 | 3 | 2 | 1 |
| Convolution Layer 2 | 32 | 64 | 3 | 2 | 1 |
| Max Pooling Layer 1 | 64 | 64 | 3 | 3 | 0 |
| Convolution Layer 3 | 64 | 128 | 3 | 2 | 1 |
| Convolution Layer 4 | 128 | 256 | 3 | 2 | 1 |
| Max Pooling Layer | 256 | 256 | 3 | 2 | 0 |
| Fully Connect Layer 1 | 1024 | 512 | - | - | - |
| Fully Connect Layer 2 | 512 | 256 | - | - | - |
| Fully Connect Layer 3 | 256 | 128 | - | - | - |

for different algorithms and different input states. For example, SAC with 4D+Cam input is trained for 324 epochs while DDPG with 4D input state is trained for 370 epochs. We train our RL models on NVIDIA GeForce RTX 3090 GPUs, and the training usually takes one day. For each trained model, we achieve a stable reward value of around 1500 for one episode.

During scenario generation, we also train a SAC model with 4D input state space as a surrogate model. The training process is the same as other models except that we use a different random seed to produce a different training result.

### A.7 Detailed scenario generation results

We show the full scenario generation and selection statistics in Table 11. We note that we don't use any personal information since our experiments are based on Carla simulation. In addition to *collision rate* (CR), *overall score* (OS), and the overall *selection rate* (SR), we also report the *average percentage of route completion* (Comp) for each scenario before and after selection to measure different algorithms' ability to influence task performances. We find that AT achieves the lowest Comp and S-Comp, which demonstrate its effectiveness in attacking the AD system's functionality.

### A.8 Full benchmark results

We report the performance of all AD algorithms tested on SafeBench in Table 12. We trained AD models with different input state spaces and evaluate their performance in both benign scenarios and safety-critical scenarios. Specifically, we provide the 4D input to all the 4 AD algorithms. For the 4D+Dir input state, we provide it to SAC, TD3, and PPO. We also equip SAC and PPO with both 4D+BEV and 4D+Cam state spaces. As shown in the table, we first notice that a large performance gap between evaluation results on benign and safety-critical scenarios always exists no matter what kind of input information we provide to the AD algorithm, which demonstrates that our testing scenarios can generalize to algorithms with different inputs. Besides, similar to the results of algorithms with 4D input, we also observe the trade-off between performance on benign and safety-critical scenarios in 4D+BEV and 4D+Cam input state spaces. For instance, when using 4D+Cam as input state space, SAC obtains a better score on benign scenarios while PPO gets a higher score on safety-critical scenarios. Finally, among different agents, PPO with 4D+BEV input achieves the best OS on SafeBench testing scenarios, which indicates potential possible directions for researchers to design their own model architecture and input state space.

Table 11: Full statistics of scenario generation and selection.

| Metric | Algo. | Traffic Scenarios: Straight Obstacle | Turning Obstacle | Lane Changing | Vehicle Passing | Red-light Running | Unprotected Left-turn | Right-turn | Crossing Negotiation | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|
| CR ↑ | LC | 0.320 | 0.140 | 0.560 | 0.920 | 0.410 | 0.630 | 0.458 | 0.470 | 0.489 |
| | AS | 0.570 | 0.350 | 0.650 | 0.900 | 0.600 | 0.820 | 0.520 | 0.550 | 0.620 |
| | CS | 0.610 | 0.630 | 0.322 | 0.900 | 0.767 | 0.756 | 0.667 | 0.711 | 0.670 |
| | AT | 0.680 | 0.310 | 0.700 | 0.930 | 1.000 | 0.850 | 0.500 | 0.900 | **0.734** |
| S-CR ↑ | LC | 0.756 | 0.923 | 0.560 | 0.919 | 0.833 | 0.870 | 0.661 | 0.793 | 0.789 |
| | AS | 0.794 | 0.595 | 0.650 | 0.900 | 0.833 | 0.930 | 0.792 | 0.797 | 0.787 |
| | CS | 0.967 | 0.684 | 0.322 | 0.900 | 0.932 | 0.870 | 0.711 | 0.797 | 0.773 |
| | AT | 0.847 | 0.485 | 0.697 | 0.930 | 1.000 | 0.966 | 0.562 | 1.000 | **0.811** |
| Comp ↓ | LC | 0.842 | 0.934 | 0.704 | 0.680 | 0.805 | 0.744 | 0.843 | 0.780 | 0.792 |
| | AS | 0.713 | 0.928 | 0.649 | 0.673 | 0.740 | 0.646 | 0.827 | 0.762 | 0.742 |
| | CS | 0.693 | 0.874 | 0.886 | 0.674 | 0.656 | 0.666 | 0.760 | 0.680 | 0.736 |
| | AT | 0.681 | 0.938 | 0.595 | 0.652 | 0.535 | 0.644 | 0.817 | 0.583 | **0.681** |
| S-Comp ↓ | LC | 0.631 | 0.559 | 0.704 | 0.679 | 0.601 | 0.647 | 0.771 | 0.631 | 0.653 |
| | AS | 0.600 | 0.884 | 0.649 | 0.673 | 0.639 | 0.595 | 0.725 | 0.655 | 0.678 |
| | CS | 0.521 | 0.866 | 0.886 | 0.674 | 0.582 | 0.614 | 0.740 | 0.640 | 0.690 |
| | AT | 0.576 | 0.905 | 0.596 | 0.652 | 0.535 | 0.594 | 0.794 | 0.536 | **0.649** |
| OS ↓ | LC | 0.765 | 0.825 | 0.613 | 0.451 | 0.755 | 0.632 | 0.630 | 0.646 | 0.665 |
| | AS | 0.654 | 0.718 | 0.577 | 0.465 | 0.659 | 0.544 | 0.599 | 0.606 | 0.603 |
| | CS | 0.629 | 0.577 | 0.738 | 0.464 | 0.569 | 0.571 | 0.520 | 0.522 | 0.574 |
| | AT | 0.600 | 0.737 | 0.557 | 0.455 | 0.460 | 0.526 | 0.607 | 0.423 | **0.546** |
| S-OS ↓ | LC | 0.565 | 0.461 | 0.613 | 0.451 | 0.533 | 0.518 | 0.528 | 0.476 | 0.518 |
| | AS | 0.548 | 0.600 | 0.577 | 0.465 | 0.535 | 0.492 | 0.451 | 0.480 | 0.518 |
| | CS | 0.465 | 0.550 | 0.738 | 0.464 | 0.483 | 0.519 | 0.496 | 0.473 | 0.524 |
| | AT | 0.523 | 0.654 | 0.558 | 0.455 | 0.460 | 0.471 | 0.574 | 0.372 | **0.508** |
| SR ↑ | LC | 0.410 | 0.130 | 1.000 | 0.990 | 0.420 | 0.690 | 0.590 | 0.580 | 0.601 |
| | AS | 0.680 | 0.420 | 1.000 | 1.000 | 0.720 | 0.860 | 0.530 | 0.640 | 0.731 |
| | CS | 0.600 | 0.760 | 1.000 | 1.000 | 0.822 | 0.856 | 0.922 | 0.878 | **0.855** |
| | AT | 0.590 | 0.330 | 0.990 | 1.000 | 1.000 | 0.870 | 0.890 | 0.900 | 0.821 |

### A.9 Full diagnostic report

In this section, we provide the diagnostic report of all AD algorithms tested on SafeBench. We evaluate different combinations of input state spaces and RL algorithms on 3 different levels of evaluation metrics. Results are shown in Table 13. We also provide an overall score for each level in Table 14. We find that PPO achieves the highest OS in most cases of input, with the highest score of $0.679$ with 4D+BEV state space. In addition, regarding the collision rate, by comparing agents with different input state spaces, we notice that AD algorithms with 4D input have the highest CR, while algorithms with 4D+BEV input get the lowest CR, which indicates that BEV is the most helpful information for AD systems to drive safely. Finally, we also observe the trade-off between functionality level metrics and safety level metrics with state spaces other than 4D, which means agents that perform well at the functionality level may not be safe regarding the safety level metrics. For example, with 4D+BEV input, PPO achieves lower CR than SAC, while its Comp is also $10.1\%$ lower than SAC. A similar phenomenon can also be found with 4D+Cam input state space.

Table 12: **The performance of all AD algorithms tested on SafeBench**. We evaluate 4 algorithms using 4 different state spaces. We report the average *overall score* (OS) on testing scenarios generated by all the 4 scenario generation algorithms with driving route variations. *Benign* indicates the performance of AD algorithms tested on normal driving scenarios. The last two columns show the OS averaged over all benign and safety-critical scenarios. Dir: 4D+Dir, BEV: 4D+BEV, Cam: 4D+Cam.

| State Space | Algo. | Traffic Scenarios | | | | | | | | Avg | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Straight Obstacle | Turning Obstacle | Lane Changing | Vehicle Passing | Red-light Running | Unprotected Left-turn | Right-turn | Crossing Negotiation | Benign | Safety-critical |
| 4D | DDPG | 0.545 | 0.526 | 0.440 | 0.501 | 0.611 | 0.444 | 0.411 | 0.507 | 0.603 | 0.498 |
| | SAC | 0.533 | 0.474 | 0.577 | 0.471 | 0.482 | 0.501 | 0.503 | 0.432 | **0.833** | 0.497 |
| | TD3 | 0.479 | 0.596 | 0.477 | 0.592 | 0.532 | 0.525 | 0.459 | 0.482 | 0.830 | 0.518 |
| | PPO | 0.761 | 0.611 | 0.426 | 0.432 | 0.755 | 0.728 | 0.605 | 0.655 | 0.819 | **0.622** |
| Dir | SAC | 0.608 | 0.591 | 0.670 | 0.435 | 0.624 | 0.548 | 0.552 | 0.522 | 0.752 | 0.569 |
| | TD3 | 0.728 | 0.543 | 0.499 | 0.451 | 0.665 | 0.595 | 0.645 | 0.590 | **0.848** | **0.590** |
| | PPO | 0.506 | 0.526 | 0.601 | 0.428 | 0.558 | 0.474 | 0.487 | 0.568 | 0.628 | 0.518 |
| BEV | SAC | 0.501 | 0.567 | 0.647 | 0.446 | 0.486 | 0.521 | 0.449 | 0.434 | **0.840** | 0.506 |
| | PPO | 0.818 | 0.632 | 0.555 | 0.393 | 0.918 | 0.664 | 0.729 | 0.847 | 0.731 | **0.694** |
| Cam | SAC | 0.634 | 0.570 | 0.436 | 0.427 | 0.481 | 0.529 | 0.527 | 0.425 | **0.812** | 0.504 |
| | PPO | 0.542 | 0.503 | 0.407 | 0.425 | 0.928 | 0.519 | 0.579 | 0.808 | 0.613 | **0.589** |

Table 13: **Diagnostic report of all AD algorithms tested on SafeBench**. We test 4 AD algorithms with 4 different state spaces on all selected testing scenarios and report the evaluation results on three different levels. Dir: 4D+Dir, BEV: 4D+BEV, Cam: 4D+Cam.

| State Space | Algo. | Safety Level | | | | Functionality Level | | | Etiquette Level | | | OS ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | CR ↓ | RR ↓ | SS ↓ | OR ↓ | RF ↑ | Comp ↑ | TS ↓ | ACC ↓ | YV ↓ | LI ↓ | |
| 4D | DDPG | 0.780 | 0.089 | 0.087 | 12.619 | 0.504 | 0.466 | 20.860 | 2.488 | 0.405 | 5.764 | 0.489 |
| | SAC | 0.829 | 0.216 | 0.146 | 3.115 | 0.882 | 0.648 | 16.827 | 1.830 | 0.704 | 2.580 | 0.499 |
| | TD3 | 0.783 | 0.231 | 0.141 | 2.535 | 0.903 | 0.670 | 17.644 | 2.680 | 1.493 | 2.545 | 0.516 |
| | PPO | 0.603 | 0.287 | 0.150 | 0.099 | 0.901 | 0.751 | 18.021 | 2.461 | 1.506 | 3.528 | **0.606** |
| Dir | SAC | 0.676 | 0.209 | 0.152 | 5.658 | 0.740 | 0.705 | 23.386 | 1.892 | 0.640 | 4.565 | 0.558 |
| | TD3 | 0.655 | 0.270 | 0.144 | 0.885 | 0.887 | 0.718 | 18.899 | 2.417 | 1.187 | 4.694 | **0.579** |
| | PPO | 0.739 | 0.045 | 0.077 | 17.607 | 0.685 | 0.534 | 21.336 | 2.911 | 0.893 | 4.875 | 0.513 |
| BEV | SAC | 0.782 | 0.229 | 0.141 | 6.057 | 0.883 | 0.674 | 17.863 | 2.952 | 1.566 | 4.448 | 0.506 |
| | PPO | 0.416 | 0.262 | 0.151 | 2.180 | 0.782 | 0.756 | 30.651 | 2.592 | 1.290 | 7.319 | **0.679** |
| Cam | SAC | 0.829 | 0.261 | 0.149 | 0.014 | 0.926 | 0.637 | 15.480 | 4.354 | 1.885 | 6.139 | 0.485 |
| | PPO | 0.600 | 0.050 | 0.127 | 15.101 | 0.708 | 0.599 | 31.914 | 2.631 | 0.827 | 6.327 | **0.576** |

### A.10 Robustness evaluation examples and visualizations

In this section, we show detailed examples and visualizations of performing diverse adversarial attacks on AD systems. In Figure 8, we provide the adversarial examples of using 3 different adversarial attacks to attack 4 different point cloud segmentation models in AD algorithms. In fig. 9, we provide the visualization results of applying 4 adversarial physical semantic perturbations and transformations to different traffic objects to attack multi-modal object detection models in AD systems.

Table 14: **Level scores for different levels of evaluation metrics.** We provide 3 different scores to sum up the safety, functionality, and etiquette levels. The weights are the same as the weights used for the overall score.

| State Space | Algorithms | Safety | Functionality | Etiquette | Overall |
|---|---|---|---|---|---|
| 4D | DDPG | 0.459 | 0.541 | 0.755 | 0.489 |
| | SAC | 0.428 | 0.750 | 0.803 | 0.499 |
| | TD3 | 0.457 | 0.759 | 0.680 | 0.516 |
| | PPO | 0.568 | 0.783 | 0.671 | **0.606** |
| Dir | SAC | 0.518 | 0.685 | 0.773 | 0.558 |
| | TD3 | 0.537 | 0.763 | 0.689 | **0.579** |
| | PPO | 0.479 | 0.621 | 0.698 | 0.513 |
| BEV | SAC | 0.450 | 0.753 | 0.629 | 0.506 |
| | PPO | 0.682 | 0.675 | 0.627 | **0.679** |
| Cam | SAC | 0.431 | 0.768 | 0.507 | 0.485 |
| | PPO | 0.565 | 0.050 | 0.693 | **0.576** |

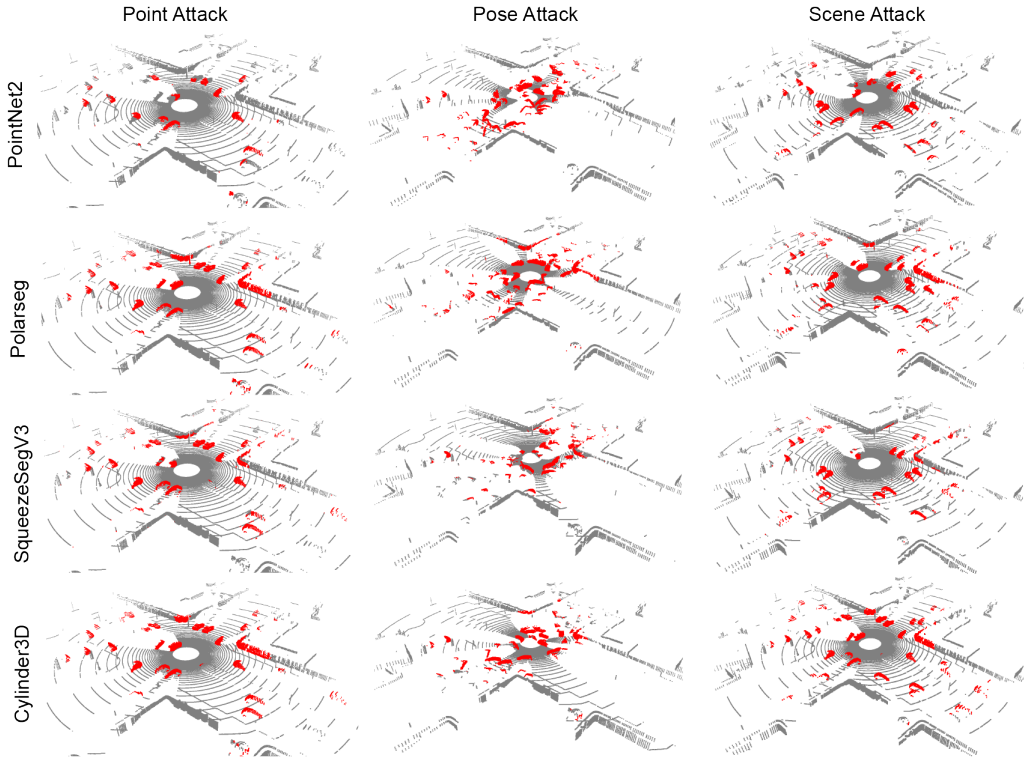



Figure 8: LiDAR point cloud of static traffic scenes generated by 3 attacking methods (Point Attack, Pose Attack, Scene Attack). Red color means the prediction of 4 point cloud segmentation algorithms.

### A.11 Complexity analysis

We provide complexity analysis of scenario generation algorithms in 2 aspects. First, for the algorithm itself, the complexity depends on the searching algorithm inside it. The 4 scenario generation algorithms considered in SafeBench all use blackbox searching algorithms, but they differ in efficiency. For example, AdvSim uses Bayesian Optimization, which improves efficiency by prioritizing hyperparameters that appear more promising from past results. Second, for the AD algorithm, the complexity depends on the robustness of the surrogate RL model. A stronger RL algorithm usually needs more optimization steps to reach our safety-critical requirements.

Figure 9: Physical semantic perturbations and transformations of vehicles and pedestrians. For vehicles, we consider different types, colors, and rotations. For pedestrians, we consider different body shapes, skin colors, and rotations.

### A.12 Potential negative societal impacts.

In SafeBench platform, we consider $8$ safety-critical scenarios and design $10$ variations for each scenario. We also systematically incorporate $4$ scenario generation algorithms with different optimization strategies to fully explore the weakness of AD algorithms. As we will open-source our platform, attackers may leverage our code and data to perform real-world adversarial attacks against existing AD systems. We suggest using our platform to evaluate the safety and robustness of AD systems in various scenarios before deploying them to the real world. Since our platform is flexible, developers and researchers can also add more safety-critical scenarios to further test and improve AD systems.