
Curriculum-based Deep Reinforcement Learning for Autonomous Driving in Simulated Highway Environments

Raahim A. Samad Poonawala
Department of Computer Science
Lahore University of Management Sciences
Lahore, 54792
27100419@lums.edu.pk

Saim Bilal
Department of Computer Science
Lahore University of Management Sciences
Lahore, 54792
27100208@lums.edu.pk

Muhammad Bin Tariq
Department of Computer Science
Lahore University of Management Sciences
Lahore, 54792
27100017@lums.edu.pk

Abstract

Inspired by CARLA [Dosovitskiy et al., 2017] and CuRLA [Uppuluri et al., 2025], we investigate curriculum-based deep reinforcement learning for autonomous driving in simulated highway environments. Our goal is to understand whether structured progression from simple to complex driving scenarios improves sample efficiency and generalization compared to training directly on complex tasks. We formulate autonomous driving as a Markov Decision Process in the OpenAI Highway Gymnasium environment and train agents using three deep RL algorithms, namely: PPO, SAC, and DQN from the Stable-Baselines3 library. Our initial experiments focus on non-curriculum baselines across four tasks: multi-lane highway driving, highway merging, signalized intersections, and roundabouts, with standardized kinematic observations and discrete meta-actions. We then design a four-stage curriculum that starts with low-traffic highway driving and gradually increases complexity through merging, intersections, and dense roundabouts under a fixed interaction budget. For Checkpoint 1, we present the problem formulation, experimental setup for both non-curriculum and curriculum regimes, and preliminary training and evaluation results on held-out scenarios.

1 Introduction

The domain of autonomous driving presents a complex safety-critical dynamic environment, where intelligent agents must be robust; able to adapt and generalize to a wide range of scenarios. Deep reinforcement learning (RL) has shown promise in training agents to make sequential decisions through trial-and-error interactions with the environment. However, training an agent to behave safely and efficiently in complex traffic scenarios can be extremely sample-inefficient and unstable.

Curriculum learning, first proposed by Bengio [Bengio et al., 2009], involves structuring the training process from simpler to more complex tasks. It proposes to address this challenge by exposing the learner to a sequence of increasingly difficult tasks rather than training directly on the hardest configuration. This approach has been proposed as a way to improve learning efficiency and generalization in RL. In the context of RL, a curriculum can be expressed as an ordered sequence

of Markov Decision Processes (MDPs) that gradually increase complexity, enabling more efficient knowledge transfer across tasks [Narvekar et al., 2020].

In this work, we investigate curriculum-based deep RL for autonomous driving in simulated highway environments, such as OpenAI’s Highway Gymnasium [Leurent, 2018] and MetaDrive [Li et al., 2022]. We aim to understand whether a structured progression through driving scenarios can lead to better performance compared to training directly on complex tasks.

2 Related Work

2.1 Curriculum Reinforcement Learning

In reinforcement learning, curriculum learning has been formalized at the level of tasks or Markov Decision Processes (MDPs). Narvekar et al. [2020] propose a general framework in which a curriculum is defined as an ordered sequence of source tasks, each represented as an MDP, and survey methods for selecting and sequencing such tasks. Their framework highlights that curricula can be used to improve sample efficiency, overcome local optima, and bridge the gap between simple training tasks and a target task that may be too difficult to learn from scratch. They also emphasize that curriculum design is largely orthogonal to the choice of underlying RL algorithm: value-based and actor–critic methods can, in principle, be combined with appropriately chosen task sequences.

Our work follows this view of curricula as ordered sequences of MDPs, instantiated in the context of autonomous driving tasks. We define a hand-crafted curriculum over four HighwayEnv scenarios that gradually increases complexity along two axes: road topology (highway, merge, intersection, roundabout) and traffic density. Unlike most prior curriculum RL work, which focuses on a single algorithm, we explicitly evaluate the interaction between curriculum design and three algorithm families: value-based off-policy control (DQN), on-policy policy-gradient actor–critic (PPO), and off-policy entropy-regularized actor–critic (SAC).

2.2 Reinforcement Learning for Autonomous Driving and Driving Simulators

A variety of driving simulators have been developed to support RL research. HighwayEnv [Leurent, 2018] is a lightweight collection of Gym-compatible environments for tactical decision-making in autonomous driving, including lane-keeping, lane-changing, merging, and intersection navigation. It provides configurable kinematic observations, discrete meta-actions, and customizable traffic conditions, making it well-suited for fast experimentation with deep RL algorithms. MetaDrive [Li et al., 2022] is a more recent simulator that emphasizes compositional scenario generation and generalization to unseen maps; it supports both single-agent and multi-agent RL tasks and has been used to study generalization, safe exploration, and multi-agent interactions.

2.3 Curriculum Learning for Autonomous Driving

Several researchers have proposed curriculum-based approaches tailored to autonomous driving. A common pattern is to fix a particular deep RL algorithm (often PPO or a related actor–critic method) and vary the environment complexity or reward shaping over training stages. For example, Uppuluri et al. [2025] introduce CuRLA, a curriculum learning framework for CARLA in which a PPO agent is combined with a variational autoencoder (VAE) to operate on compressed visual observations. They design a two-fold curriculum that gradually increases scenario difficulty and tightens safety constraints via collision penalties, reporting improvements in both training performance and evaluation robustness relative to non-curriculum baselines.

In contrast, we focus on HighwayEnv as a lightweight yet diverse testbed that includes multi-lane highways, merges, intersections, and roundabouts. We design a four-stage curriculum that starts with low-traffic highway driving and progresses through increasingly complex tasks and traffic conditions, under a fixed interaction budget. We instantiate this curriculum with three different deep RL algorithms (DQN, PPO, and SAC) and compare them against a non-curriculum baseline that trains on a fixed mixture of tasks.

3 Mathematical Formulation

We formulate the autonomous driving problem as a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where:

- \mathcal{S} is the state space, representing the kinematic observations of the ego vehicle and surrounding traffic. Each state $s \in \mathcal{S}$ includes features such as position, velocity, acceleration, lane information, and distances to nearby vehicles.
- \mathcal{A} is the action space, consisting of discrete meta-actions that the agent can take. These actions include lane changes (left, right, maintain lane) and speed adjustments (accelerate, decelerate, maintain speed).
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the state transition probability function, defining the dynamics of the environment. It specifies the probability of transitioning to state s' from state s after taking action a . Note that for our purposes, we do not require explicit knowledge of P , as we employ model-free reinforcement learning algorithms. Thus, the simulator implicitly defines these transition dynamics through its interactions.
- $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, which provides feedback to the agent based on its actions. The reward structure is designed to encourage safe and efficient driving behaviors, such as maintaining a desired speed, avoiding collisions, and adhering to traffic rules.
- $\gamma \in [0, 1]$ is the discount factor, which determines the importance of future rewards in the agent’s decision-making process.

The objective of the agent is to learn a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maximizes the expected cumulative discounted reward:

$$J(\pi) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]$$

where the expectation is taken over the trajectories generated by following policy π .

As Narvekar et al. [2020] described, there are three main classes of methods to learn an optimal policy π^* :

- Value-based methods: These methods focus on estimating the optimal action-value function $Q^*(s, a)$, which represents the expected cumulative reward of taking action a in state s and following the optimal policy thereafter. The optimal policy can then be derived by selecting the action that maximizes $Q^*(s, a)$ for each state s .
- Policy-based methods: These methods directly parameterize the policy $\pi_{\theta}(a|s)$ and optimize the policy parameters θ to maximize the expected cumulative reward $J(\pi_{\theta})$ using gradient ascent techniques.
- Actor-critic methods: These methods combine value-based and policy-based approaches by maintaining both a policy (the actor) and a value function (the critic). The critic estimates the value function, which is used to update the policy.

For the purposes of this study, we chose three representative algorithms from these classes: Deep Q-Networks (DQN) as a value-based method, Proximal Policy Optimization (PPO) as a policy-based method, and Soft Actor-Critic (SAC) as an actor-critic method. Each of these algorithms were implemented using the Stable-Baselines3 library [Raffin et al., 2021] and trained on the HighwayEnv [Leurent, 2018] simulator. For the MetaDrive [Li et al., 2022] simulator, we only used the PPO and DQN algorithms due to computational constraints.

4 Experimental Setup

In this section we describe our experimental setups in the OpenAI Highway Gymnasium environment (HighwayEnv) and in the MetaDrive simulator. HighwayEnv serves as our primary testbed for comparing curriculum and non-curriculum training under a fixed interaction budget, while MetaDrive provides a more complex, procedurally generated environment used to design a complementary curriculum and per-stage non-curriculum baselines.

4.1 HighwayEnv Experiments

4.1.1 Tasks and Environment Configuration

We use the highway-env suite of Gymnasium-compatible tactical driving tasks as our main benchmark. Specifically, we consider four scenarios:

- highway-v0: multi-lane highway driving.
- merge-v0: highway with an on-ramp merge.
- intersection-v0: signalized intersection.
- roundabout-v0: circular roundabout with entering and exiting traffic.

All tasks share a common base configuration. Episodes have a fixed duration of 40 s, with simulation and control frequencies set to 15 Hz and 5 Hz, respectively. Traffic density is controlled by a scalar parameter $\rho \in [0, 1]$, and the number of other vehicles in the scene is set to `vehicles_count = $\lfloor 50\rho \rfloor$` . We adjust ρ per task to create a range of difficulties, and slightly perturb it between training and test for held-out evaluation.

4.1.2 Observation and Action Spaces

To standardize the input across tasks, we use the built-in `Kinematics` observation:

$$s_t = \text{concat} \left[(\text{presence}_i, x_i, y_i, v_{x,i}, v_{y,i})_{i=0}^{N-1} \right],$$

where $N = 5$ is the number of vehicles (ego plus nearest neighbors), and the features are normalized and expressed in relative coordinates. This yields a fixed-dimensional, low-level kinematic state vector.

The action space is discrete and defined via `DiscreteMetaAction`, which maps a small set of high-level actions (e.g., keep lane, change lane left/right, accelerate, brake) to continuous control commands for the underlying vehicle dynamics. This choice allows us to apply both value-based (DQN) and policy-gradient (PPO, SAC) methods without changing the environment.

4.1.3 Algorithms and Hyperparameters

We instantiate three reinforcement learning algorithms from Stable-Baselines3: PPO, A2C, and DQN, each using a feed-forward MLP policy over the input observations.

PPO. We use an `MlpPolicy` with two hidden layers of 256 units each, employing the Tanh activation function for better performance on normalized observations. The learning rate is set to 5×10^{-4} , with a batch size of 64 and a rollout length of 2048 steps. We run 10 optimization epochs per update, and the clipping parameter for PPO is set to 0.2. The discount factor is $\gamma = 0.99$, and we use Generalized Advantage Estimation (GAE) with a λ of 0.95. The entropy coefficient is set to 0.01 to encourage exploration, and the value function coefficient is set to 0.5. Gradient clipping is applied with a maximum gradient norm of 0.5. State-dependent exploration is disabled by setting `use_sde` to False, and we normalize the advantages during training.

A2C. We use an `MlpPolicy` with two hidden layers of 256 units each, similar to PPO, but with slightly different hyperparameters. The learning rate is set to 7×10^{-4} , with a smaller rollout length of 8 steps and 10 optimization epochs per update. The discount factor remains $\gamma = 0.99$, and GAE is turned off with a λ value of 1.0, as A2C typically uses Monte Carlo methods for advantage estimation. The entropy coefficient is set to 0.01 to promote exploration, and the value function coefficient is set to 0.25. Gradient clipping is applied with a maximum gradient norm of 0.5. We use RMSprop as the optimizer with a small epsilon value of 1×10^{-5} , and advantage normalization is enabled.

DQN. We use an `MlpPolicy` with two hidden layers of 256 units each, utilizing the ReLU activation function, which is effective for Q-learning tasks. The learning rate is set to 10^{-4} , with a larger replay buffer size of 100,000 and a batch size of 32. The target network is updated every 1000 steps, and we use a hard update for the target network with $\tau = 1.0$. The exploration fraction is set to 0.1, with an initial ϵ of 1.0 and a final ϵ of 0.05, using an ϵ -greedy exploration schedule. The discount factor is $\gamma = 0.99$, and the training frequency is every 4 environment steps. Gradient steps are performed once per update, and we disable memory optimization for compatibility.

4.2 Training Protocols

4.2.1 Non-Curriculum Baseline

In the non-curriculum baseline, the agent trains on a fixed mixture of four tasks: `highway-v0`, `merge-v0`, `intersection-v0`, and `roundabout-v0`, each with a specified traffic density. Conceptually, this corresponds to sampling an MDP $\mathcal{M} \sim p(\mathcal{M})$ from a distribution over these tasks and running standard RL.

In practice, our implementation cycles through the four tasks, allocating an equal share of a fixed interaction budget (e.g., 25,000 timesteps per task for a total of 100,000 timesteps), and training a single policy across all of them.

For each algorithm, we:

1. Initialize a policy with random parameters.
2. For each of the four tasks, train the policy for $T/4$ timesteps (e.g., 25,000) using vectorized environments with n_{envs} parallel instances.
3. After each task-specific training block, evaluate the current policy on a held-out configuration of the same task for a fixed number of episodes (e.g., 10) and record the mean and standard deviation of episode returns.

We log training and evaluation results to disk, including checkpoint models and summary JSON files.

4.2.2 Curriculum Regimen

In the curriculum setting, we define a four-stage curriculum that progresses through the tasks in increasing order of complexity (defined as an MDP sequence):

$$\mathcal{M}_1 = \text{highway-v0} \rightarrow \mathcal{M}_2 = \text{merge-v0} \rightarrow \mathcal{M}_3 = \text{intersection-v0} \rightarrow \mathcal{M}_4 = \text{roundabout-v0}$$

where each stage uses a progressively higher traffic density and represents a more complex driving scenario. We allocate a fixed interaction budget T across the four stages, training the agent sequentially on each MDP for $T/4$ timesteps.

We initialize the policy parameters at Stage 1 and train for T_{stage} steps using the stage-specific environment. At the end of each stage we (i) evaluate the current policy on that stage’s environment and record mean and standard deviation of returns, and (ii) save a stage-specific checkpoint. The model parameters are then carried forward unchanged to the next stage. After completing all four stages, we evaluate the final policy on the same held-out test configurations $\mathcal{M}_{\text{test}}$ used in the non-curriculum setting.

This protocol allows us to compare curriculum and non-curriculum training under identical sample budgets and observation/action spaces, while varying only the order and structure of the encountered tasks.

4.2.3 Evaluation Metrics (HighwayEnv)

Our primary evaluation metric is the average undiscounted episode return on held-out configurations:

$$\bar{R} = \frac{1}{N} \sum_{i=1}^N R^{(i)},$$

where $R^{(i)}$ is the total reward in evaluation episode i and N is the number of evaluation episodes. When available from the environment, we also log auxiliary metrics such as collision rate, success/completion rate, and average episode length.

4.3 MetaDrive Experiments

MetaDrive is a more complex, procedurally generated driving simulator with diverse road layouts and traffic patterns. In this project, we use it to (i) design a complementary curriculum and (ii) train per-stage non-curriculum baselines that we evaluate on challenging held-out scenarios. Due to computational constraints, we restrict MetaDrive experiments to PPO and DQN.

4.3.1 Curriculum Stage Design (MetaDrive)

We define four curriculum stages $\{C0, C1, C2, C3\}$ with increasing geometric and traffic complexity. Each stage has its own interaction budget and reward-shaping parameters:

- **C0 (Straight, no traffic).** Single straight road segment (map "S") with no other vehicles; budget 100,000 steps. The reward mainly encourages staying on the road and maintaining moderate speed.
- **C1 (Roundabout, no traffic).** Single roundabout (map "0") with no traffic; budget 150,000 steps. Topology is more complex than C0, but the ego vehicle still drives alone.
- **C2 (Light-traffic PG map).** A procedurally generated 10-block map with light traffic (e.g., density $\rho \approx 0.05$); budget 200,000 steps. The reward increases emphasis on speed and safety, penalizing collisions and traffic violations more strongly.
- **C3 (Dense-traffic PG map).** A 20-block map with dense traffic (e.g., $\rho \approx 0.30$); budget 200,000 steps. This is the most challenging stage, with a greater success bonus and higher penalties (compared to other stages).

The total MetaDrive curriculum budget is the sum of the four stage budgets. In addition, we also add a progressively larger step penalty for each stage which encourages the ego vehicle to reach its destination faster

4.3.2 Environment Configuration and Reward Shaping

For each stage, we instantiate a MetaDrive environment with the specified map and traffic level and wrap it in a custom CurriculumRewardWrapper. This wrapper combines:

- a base term proportional to the original environment reward,
- a speed term that rewards driving up to a stage-specific speed limit,
- penalties for collisions, off-road events, and traffic violations,
- a terminal success bonus for safe arrival at the destination,
- and a small per-step penalty to discourage unnecessarily long episodes.

For DQN, we apply a discrete-action wrapper that maps a small set of steering–throttle pairs to discrete actions; PPO operates directly in the continuous action space.

4.3.3 Non-Curriculum Baselines (MetaDrive)

At this checkpoint, we use the stages above to define per-stage non-curriculum baselines. For each stage C_k and each algorithm (PPO, DQN), we:

1. train a separate model from scratch on that stage only, using the stage’s budget as the total number of environment steps, and
2. evaluate the trained policy on the same stage.

For each run, we log average return, and episode length. These stage-wise baselines will serve as reference points for future curriculum experiments in MetaDrive.

4.3.4 Held-out Test Scenarios (MetaDrive)

To assess generalization, we additionally evaluate each trained policy on two held-out scenarios that are never seen during training:

- **Composed Map Scenario.** A fixed six-block map (denoted "SCrRX0") consisting of straight, circular, ramp, intersection, and roundabout segments, with medium traffic. This tests transfer to a more complex but structured layout.
- **Varying Dynamics Scenario.** A map with randomized vehicle dynamics and a distribution over traffic and physical parameters (VaryingDynamicsEnv), which tests robustness to changes in dynamics and model mismatch.

For each held-out scenario, we run a fixed number of evaluation episodes and log the same metrics as above. Overall, HighwayEnv and MetaDrive together provide a progression from lightweight, fast-to-train tactical tasks to richer, procedurally generated environments. At this checkpoint we focus on non-curriculum baselines and curriculum design; subsequent checkpoints will execute full curriculum-versus-non-curriculum comparisons in both environments.

5 Preliminary Results

In this section, we will summarize and evaluate performance across the two benchmarks (MetaDrive and OpenAI Gym’s HighwayEnv) for their respective algorithms. For MetaDrive, we performed non-curriculum training and testing across three algorithms: Deep Q-Networks, Soft Actor-Critic, and Proximal Policy Optimization. For HighwayEnv, we performed both curriculum and non-curriculum training and testing for DQN and PPO. It must also be noted that for HighwayEnv, we did not utilize SAC due to training convergence issues and so, had to resort to Advantage Actor-Critic (A2C) which unfortunately failed in curriculum training and testing.

5.1 MetaDrive Non-curriculum Results and Evaluation

Training was done for 4 set-ups:

1. Straight road with zero traffic
2. Roundabout with zero traffic
3. 20-block map with low traffic
4. 20-block map with dense traffic

Testing was then conducted on a held-out map (SCrRXO) with medium traffic and the following topology: straight -> circular -> in-ramp -> out-ramp -> intersection -> roundabout. Testing followed after training on each of the 4 set-ups.

5.1.1 Plots

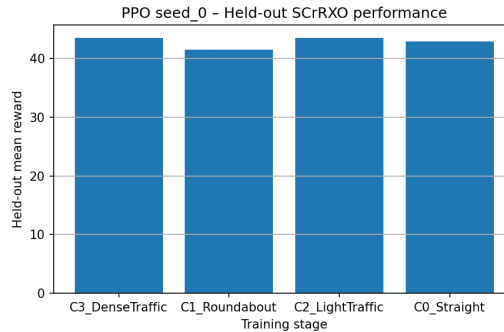


Figure 1: Held-out Map Performance for PPO-based Training

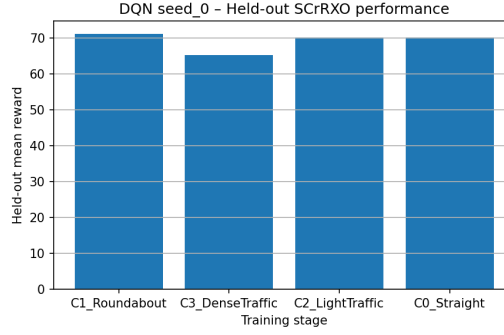


Figure 2: Held-out Map Performance for DQN-based Training

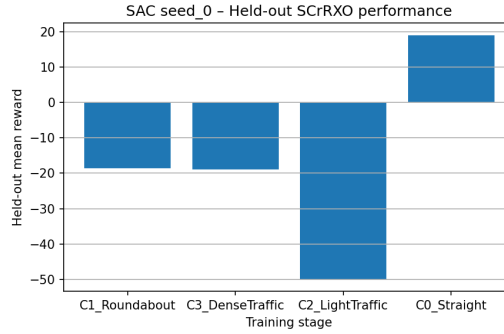


Figure 3: Held-out Map Performance for SAC-based Training

5.1.2 Analysis

DQN and PPO performed almost identically on all three held-out evaluation maps across the four training-map variations, indicating that both algorithms learned stable, map-agnostic driving strategies rather than memorizing particular layouts.

SAC, on the other hand, was obviously sensitive to the training environment. SAC demonstrated its best held-out performance when trained on the straightforward straight-road map; however, after training on more complicated roundabout or traffic-heavy maps, its returns significantly decreased. This pattern suggests that SAC tended to overfit to the local dynamics of complex scenarios, resulting in policies that were less adaptable to unseen road structures but more effective in-distribution.

5.2 HighwayEnv Results and Evaluation

Training was done on the following 3 maps:

1. highway-v0 with 20% traffic
2. merge-v0 with 30% traffic
3. roundabout-v0 with 25% traffic

Testing was done on intersection-v0 with variable traffic. For non-curriculum, testing proceeded after training on each of the three stages. For curriculum, testing proceeded once at the end after obtaining the final model post a three-stage long training.

5.2.1 Plots

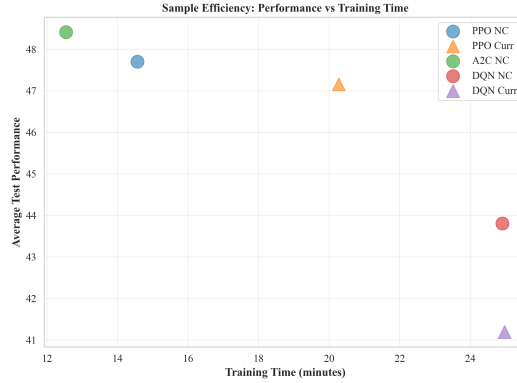


Figure 4: Testing Performance and Training Time

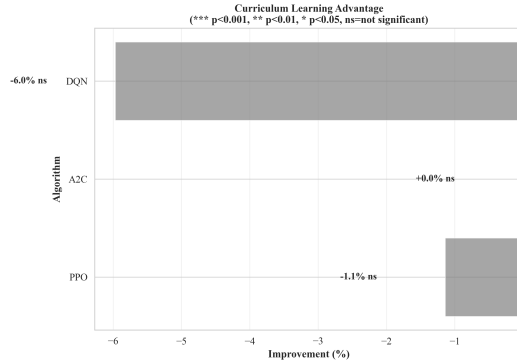


Figure 5: Statistical Significance

5.2.2 Analysis

Contrary to our initial expectation that progressively increasing task difficulty would improve learning stability and sample-efficiency, our results on HighwayEnv show no measurable benefit from curriculum training for either DQN or PPO. Across all training seeds, non-curriculum runs achieved slightly higher average returns than their curriculum counterparts, and the training times for non-curriculum trainings were even lower as well. Statistical analysis supports this observation: Cohen’s d-values for curriculum vs. non-curriculum were negligible, and two-sample significance tests yielded non-significant differences, indicating that the observed performance gap is not meaningful. Overall, within our experimental setup, curriculum training did not improve, and in some cases slightly reduced, performance, suggesting that HighwayEnv may not strongly benefit from curriculum design or that our specific curriculum schedule did not introduce helpful intermediate structure.

References

- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09)*, pages 41–48. ACM / Machine Learning Research, 2009. doi: 10.1145/1553374.1553380.
- Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator, 2017. URL <https://arxiv.org/abs/1711.03938>.
- Edouard Leurent. An environment for autonomous driving decision-making. <https://github.com/eleurent/highway-env>, 2018.

- Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey, 2020. URL <https://arxiv.org/abs/2003.04960>.
- Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL <http://jmlr.org/papers/v22/20-1364.html>.
- Bhargava Uppuluri, Anjel Patel, Neil Mehta, Sridhar Kamath, and Pratyush Chakraborty. Curla: Curriculum learning based deep reinforcement learning for autonomous driving, 2025. URL <https://arxiv.org/abs/2501.04982>.