C2SMARTER

CONNECTED COMMUNITIES WITH SMART MOBILITY TO EQUITABLY REDUCE CONGESTION

A U.S. DOT University Transportation Center

New York University
NYC College of Technology
North Carolina A&T University
Rutgers University
Texas Southern University
University of Texas at El Paso
University of Washington

Building Digital Twins via GPU-Accelerated Human Regularized Reinforcement Learning

September 2024

Building Digital Twins via GPU-Accelerated Human-Regularized Reinforcement Learning

Eugene Vinitsky
NYU
0000-0003-2372-4944

Daphne Cornelisse
NYU
0009-0007-7801-6095

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
4. Building Digital Twins via GPU-Accelerated Human-Regularized Reinforcement Learning		5. Report Date September 2024
		6. Performing Organization Code:
7.Author(s) Eugene Vinitsky, Daphne Cornelisse		8. Performing Organization Report No.
9. Performing Organization Name and Address Connected Cities for Smart Mobility towards Accessible and Resilient Transportation for Equitably Reducing Congestion Center (C2SMARTER) 6 Metrotech Center, 4th Floor NYU Tandon School of Engineering Brooklyn, NY, 11201, United States		10. Work Unit No.
		11. Contract or Grant No. 69A3552348326
12. Sponsoring Agency Name and Address Office of Research, Development, and Technology Federal Highway Administration 6300 Georgetown Pike McLean, VA 22101-2296		13. Type of Report and Period Final report 09/30/2023-09/30/2024
		14. Sponsoring Agency Code
15. Supplementary Notes		

16.Abstract

We have carried out a two-component study investigating how to build high-performing human driver models. First, we investigated how to combine large-scale datasets from autonomous vehicle companies with reinforcement learning methods for training agents. We showed that by adding a supervised learning loss atop the model training, we could build models that were human-like without reducing the agent performance. However, we were bottlenecked by simulator speed. We then designed a new simulator, GPUDrive, that can run thousands of simultaneous simulations of urban environments containing drivers, pedestrians, and cyclists. This simulator is an order of magnitude faster than existing open-source simulators. We demonstrate that in this simulator, we can reproduce our prior agent training results between one and two orders of magnitude faster.

17. Key Words		18. Distribution Statement No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA 22161. http://www.ntis.gov	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages	22. Price

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

Disclaimer

Acknowledgements

This was supported in part through the NYU-IT High-Performance Computing resources, services, and staff expertise.

Executive Summary

Modeling human drivers is a central challenge for estimating the impacts of congestion reduction, making the incorporation of realistic human-like agents essential for scalable training and evaluation in simulation. However, applying multi-agent reinforcement learning (MARL) techniques to modeling human driving has been hindered by the requirement of billions of steps of experience and the high collision rates of pure imitation learning agents in closed-loop settings. To address these challenges, we present **GPUDrive**, a GPU-accelerated multi-agent simulator built on top of the Madrona Game Engine, capable of generating over a million steps of experience per second. This high-performance environment allows for the efficient training of complex, heterogeneous agents, with observation, reward, and dynamics functions written directly in C++ and compiled to high-performance CUDA code.

Using GPUDrive to train the agents, we propose **Human-Regularized Proximal Policy Optimization (HR-PPO)**, a multi-agent algorithm where agents are trained through self-play with a small penalty for deviating from a human reference policy. In contrast to prior imitation-based approaches, our RL-first method utilizes only 30 minutes of imperfect human demonstrations. Training over numerous scenes from the Waymo Motion dataset, we achieve highly effective goal-reaching agents within minutes for individual scenes and generally capable agents in a few hours. Empirical evaluations demonstrate that our HR-PPO agents attain a success rate of 93%, an off-road rate of 3.5%, and a collision rate of 3% across a wide array of multi-agent traffic scenarios. Furthermore, the agents exhibit human-like driving behaviors, closely matching existing human driving logs, and show significant improvements in coordinating with human drivers, particularly in highly interactive situations.

We open-source GPUDrive and our trained agents at https://github.com/Emerge-Lab/gpudrive. Demonstrations of agent behaviors are available at https://sites.google.com/view/driving-partners, facilitating further research and development in scalable multi-agent autonomous driving simulation.

Table of Contents

List of Figures

Introduction

Developing autonomous vehicles (AVs) that are compatible with human driving remains a challenging task, especially given the low margin for error in the real world. Driving simulators offer a cost-effective and safe means to develop and refine autonomous driving systems. The purpose of these simulators is to prepare AVs for real-world deployment, where they must smoothly interact and coordinate with a diverse set of human drivers. Therefore, a crucial aspect of both learning and validation in these simulators involves realistic simulations: the traffic scenarios and other simulation agents with which the controlled AV interacts. To identify where driving policies fall short, it is important to ensure that the simulated traffic conditions and driver agents closely resemble those in the real world [1, 2].

Existing driving simulators typically provide a set of baseline agents to interact with, such as low-dimensional car following models, rule-based agents, or recorded human driving logs [3,1,4]. While these agents provide a form of interactivity, they are limited in their abilities to create interesting and challenging coordination scenarios, which requires driving agents that are reactive and sufficiently human-like. Having effective simulation agents that drive and respond in human-like ways would facilitate the controlled generation of human-AV interactions, which has the potential to unlock realistic training and evaluation in simulation at scale. Additionally, it would reduce the need for continuous real-world large-scale data collection.

Building human-like driving policies is an ongoing challenge. Existing simulated agents are either (1) quite far from human-like behavior (2) struggle with achieving closed-loop stability or (3) frequently get stuck in deadlocks. A ubiquitous way to generate driving policies has been through imitation learning, where a driving policy is learned by mimicking expert behavior using recorded actions from human drivers [5,6] Unfortunately, such policies still have high crash rates when put in a multi-agent closed-loop setting where they have to respond to the actions of other agents [7]. Another approach that has been explored to achieve closed-loop stability is multi-agent RL [8]. While in principle perfect closed-loop driving may be achieved via self-play, there is no guarantee that the equilibrium the agents find will be at all human-like. For example, self-play agents have no a priori reason to prefer driving on the left side of the road vs. the right. Similarly, because every agent is aware that other agents are a copy of themselves, they may feel comfortable driving much closer to each other than human comfort and reaction times would allow. Resolving this requires some mechanism of specifying what it means for a policy to be "human-like" and training agents to match this specification. This is the challenge we tackle in this work.

Subsection 1.1 – GPUDrive:

To address the challenges of driving and unlock multi-agent learning as a tool for generating capable self-driving planners, we introduce GPUDrive. GPUDrive is a simulator intended to mix real-world driving data with simulation speeds that enable the application of sample-inefficient but effective RL algorithms to planner design. GPUDrive runs at over a million steps per second on both consumer-grade and datacenter-class GPUs and has a sufficiently light memory footprint to support hundreds to thousands of simultaneous worlds (environments) with hundreds of agents per world. GPUDrive supports the simulation of a variety of sensor modalities, from LIDAR to a human-like view cone, enabling GPUDrive to be used for studying the effects of different sensor types on resultant agent characteristics. Finally, GPUDrive takes in driving logs and maps from existing self-driving datasets, enabling the mixing of tools from imitation learning with reinforcement learning algorithms. This enables the study of both the development of autonomous vehicles and the learning of models of human driving, cycling, and walking behavior.

Subsection 1.2 – Human-Regularized PPO:

In the latter half of the work we address the challenge of developing human-like drivers with GPUDrive. we propose Human-Regularized PPO (HR-PPO). HR-PPO is an on-policy algorithm that includes an additional regularization term that nudges agents to stay close to human-like driving. Concretely, our contributions are:

- We show that adding a regularization term to PPO agents trained in self-play leads to agentsthat are more compatible with proxies for human behavior in a variety of scenarios in GPUDrive.
- Our results also show that effectiveness (being able to navigate to a goal without colliding) and realism (driving in a human-like way) can be achieved simultaneously: Our HR-PPO agents achieve similar performance to PPO while experiencing substantial gains inhuman-likeness.
- We also show the benefits of training in multi-agent settings: HR-PPO self-play agents outperform agents trained directly on the test distribution of agents. This suggests that multi-agent training may provide additional benefits over single-agent training (log-replay).

Literature Review

Subsection 2.1: Driving agents in simulation

There are four major approaches used in existing traffic simulators to model human drivers. One class of methods uses **low-dimensional car following models** to describe the dynamics of vehicle movement through a small number of variables or parameters [9, 10, 3]. **Rule-based agents** have a fixed set of behaviors. Examples of rule-based agents in driving simulators include car-following agents [1, 11, 12, 13} such as the IDM model and behavior agents that can be parameterized to drive more cautiously or aggressively such as CARLA's TrafficManager [4]. While car-following and rule-based agents can respond to other agents and thus provide interactivity, it can be challenging for them to capture the full complexity of human driving behavior and these agents frequently experience non-physical accelerations or come to a deadlock in complex interactions. Some simulators provide **human driving logs** which can be replayed to allow for interactions [14, 8, 1, 15, 11]. Although these static models produce realistic trajectories, they cannot respond to changes in the environment, such as other drivers. Finally, some driving simulators include l**earning-based agents** using reinforcement learning [16] however, these agents likely do not resemble human behavior. Our Human-Regularized PPO approach aims to produce simulation agents that meet all these criteria to allow for the controlled generation of challenging real-life interactions in simulation.

Subsection 2.2: Imitation Learning and Supervised Learning.

A canonical approach for developing learning-based driving policies for autonomous driving has been through Imitation Learning (IL) [5, 17, 6] and other supervised methods such as trajectory prediction [18] and language-conditioned traffic scene generation [19]. IL works by mimicking expert behavior using recorded actions from human drivers. There are two broad classes of IL: *open-loop* and *closed-loop*. Open-loop methods, like Behavioral Cloning (BC), learn a policy without taking into account real-time feedback. As such, one limitation of open-loop IL methods is that they suffer from compounding errors once deployed in closed-loop systems [20]. Closed-loop IL[21, 22, 23, 24, 25, 26] improves upon this by letting the system adjust its actions through ongoing interaction with the environment during training. While these methods provide enhanced robustness, they have not yet achieved high closed-loop performance when all agents are controlled. In addition, our approach does not rely on large, high-quality datasets of human driving data.

Subsection 2.3: Multi-Agent Reinforcement Learning

Reinforcement learning techniques have been effective in developing capable agents without requiring human data [27, 28, 29] in zero-sum and collaborative games. While this approach has worked in a range of games [30, 31] many games have multiple equilibria such that agents trained in self-play do not perform well when matched with human-partners [33, 34].

In the driving setting, this challenge can partly be ameliorated through the design of reward functions that encode how people drive and behave in traffic interactions [36, 37]. However, it is not entirely clear what reward function corresponds to human driving and the inclusion of this type of reward shaping can create undesired behaviors [32]. An alternate approach tries to create human compatibility through the design of training procedures that restrict the set of possible equilibria [34, 35] by ruling out equilibria that humans are unlikely to play.

Subsection 2.4: Combined IL + MARL

Recent work has shown that augmenting IL with penalties for driving mistakes can create more reliable policies. This has been demonstrated in both closed-loop [38, 39] and open-loop [14] settings. Outside of the driving domain, augmenting goal-conditioned single-agent reinforcement learning has been found to enhance performance in the Arcade Learning Environment (ALE) [40] and improve the likelihood of convergence to the equilibrium in certain multi-agent learning settings [41, 42]. In multi-agent settings, it has empirically been shown to yield policies more compatible with existing social conventions of the human reference group [43, 44, 33] Our approach extends these works to the driving setting where it has not yet been investigated in prior work if this type of data-driven regularization is sufficient to enable convergence to a human-compatible policy.

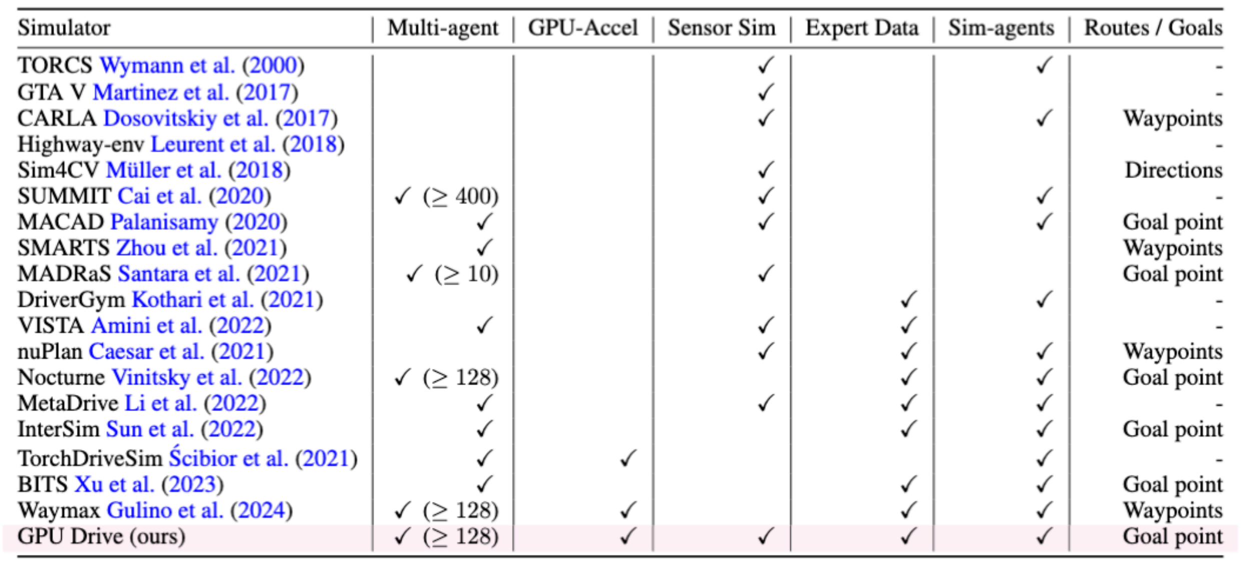
Subsection 2.5: Frameworks for batched simulators.

There are various open-source frameworks available that support hardware-accelerated reinforcement learning environments. These environments are generally written directly in an acceleration framework such as Numpy [50] Jax [66] or Pytorch [48]

In terms of multi-agent accelerated environments, standard benchmarks include JaxMARL [70], Jumanji [69] and VMAS [47] which primarily feature fully cooperative or fully competitive tasks. Each benchmark requires the design of custom accelerated structures per environment.

In contrast, GPUDrive focuses on a mixed motive setting and is built atop Madrona, an extensible ECS-based framework in C++, enabling GPU acceleration and parallelization across environments [71]. Madrona comes with vectorization of key components of embodied simulation such as collision checking and sensors such as LIDAR. GPUDrive can support hundreds of controllable agents in more than 100,000 distinct scenarios, offering a distinct generalization challenge and scale relative to existing benchmarks. Moreover, GPUDrive includes a large dataset of human demonstrations, enabling imitation learning, inverse RL, and combined IL-RL approaches.

Subsection 2.6: Simulators for autonomous driving research and development



Simulator	Multi-agent	GPU-Accel	Sensor Sim	Expert Data	Sim-agents	Routes / Goals
TORCS Wymann et al. (2000)			✓		✓	-
GTA V Martinez et al. (2017)			✓			-
CARLA Dosovitskiy et al. (2017)			✓		✓	Waypoints
Highway-env Leurent et al. (2018)						-
Sim4CV Müller et al. (2018)			✓			Directions
SUMMIT Cai et al. (2020)	✓ ($\geq$ 400)		✓		✓	-
MACAD Palanisamy (2020)	✓		✓		✓	Goal point
SMARTS Zhou et al. (2021)	✓					Waypoints
MADRaS Santara et al. (2021)	✓ ($\geq$ 10)		✓			Goal point
DriverGym Kothari et al. (2021)				✓	✓	-
VISTA Amini et al. (2022)	✓		✓	✓		-
nuPlan Caesar et al. (2021)			✓	✓	✓	Waypoints
Nocturne Vinitsky et al. (2022)	✓ ($\geq$ 128)			✓	✓	Goal point
MetaDrive Li et al. (2022)	✓		✓	✓	✓	-
InterSim Sun et al. (2022)	✓			✓	✓	Goal point
TorchDriveSim Ścibior et al. (2021)	✓	✓			✓	-
BITS Xu et al. (2023)	✓			✓	✓	Goal point
Waymax Gulino et al. (2024)	✓ ($\geq$ 128)	✓		✓	✓	Waypoints
GPU Drive (ours)	✓ ($\geq$ 128)	✓	✓	✓	✓	Goal point

Figure 1: Comparison of different simulators with GPUDrive. Multi-agent column refers to whether the simulator supports multi-agent control, GPU-Accel to whether the simulator is GPU accelerated, Sensor Sim to whether it provides implementations of sensors like cameras or LIDAR, Expert Data to whether it has human driver data, Sim-agents to whether it comes with default actor models, and Routes / Goals to the type of navigation information provided to the agents.

The purpose of GPUDrive is to facilitate the systematic study of behavioral, coordination, and control aspects of autonomous driving and multi-agent learning more broadly. As such, visual complexity is reduced, which differs from several existing simulators, which (partially) focus on perception challenges in driving [4, 65] Driving simulators close to GPUDrive in terms of either features or speed include MetaDrive [16] nuPlan [11], Nocturne [8], and Waymax [1] which all utilize real-world data. Unlike MetaDrive and nuPlan, our simulator is GPU-accelerated. Like GPUDrive, Waymax is a JAX-based GPU-accelerated simulator that achieves high throughput through JIT compilation and efficient use of accelerators. With respect to Waymax, our simulator supports a wider range of possible sensor modalities including LIDAR and human-like views, can scale to nearly thirty times more worlds, and comes with performant reinforcement learning baselines. However, it does not currently come with reactive IDM agents like Waymax though it does come with pre-trained simulated agents based on RL policies.

Data and Methodology

Subsection 3.1: Simulator design

Learning to safely navigate complex scenarios in a multi-agent setting requires generating many billions of environment samples. To feed sample-hungry learning algorithms, GPUDrive is built on top of Madrona [71], an Entity-Component-State system designed for high-throughput reinforcement learning environments. In the Madrona framework, multiple independent worlds (each containing an independent number of agents[1]) are executed in parallel on accelerators via a shared engine.

However, driving simulation offers a particular set of challenges that require several technical choices. First, road objects, such as road edges and lane lines are frequently represented as polylines (i.e. connected sets of points). These polylines can consist of hundreds of points as they are sampled at every 0.1 meters, leading to even small maps having upwards of tens of thousands of points. This can blow up the memory requirements of each world as well as lead to significant redundancy in agent observations. Second, the large numbers of agents and road objects can make collision checking a throughput bottleneck. Finally, there is immense variability in the number of agents and road objects in a particular scene. Each world allocates memory to data structures that track its state and accelerate simulation code. Though independent, each world incurs a memory footprint proportional to the *maximum* number of agents across all worlds. In this way, the performance of GPUDrive is sensitive to the variation in agent counts across all the worlds in a batch.

These challenges are partially resolved via the following mechanisms. First, a primary acceleration data structure leveraged by GPUDrive is a Bounding Volume Hierarchy (BVH). The BVH keeps track of all physics entities and is used to easily exclude candidate pairs for collisions. This allows us to then run a reduced-size collision check on potential collision candidate pairs. The use of a BVH avoids invoking a collision check that would otherwise always be quadratic in the number of agents in a world. Secondly, we observed that a lot of the lines in the geometry of the roads are straight. This allows us to omit many intermediate points while only suffering a minor hit in the quality of the curves. We apply a polyline decimation algorithm (Viswalingham-Whyatt Algorithm) to approximate straight lines and filter out low-importance points in the polylines. With this modification, we can reduce the number of points by 10-15 times and significantly improve the step times while decreasing memory usage. Finally, rather than allocate memory for the maximum number of agents in a scene (as is likely necessary in frameworks like Jax), we only allocate memory equal to the actual number of instantiated agents.

We provide an overview of some of the pertinent simulator features as well as sharp edges and limitations of the simulator as a guide to potential users.

[1] In the Waymo Open Motion Dataset, an agent constitutes a vehicle, cyclist, or pedestrian.

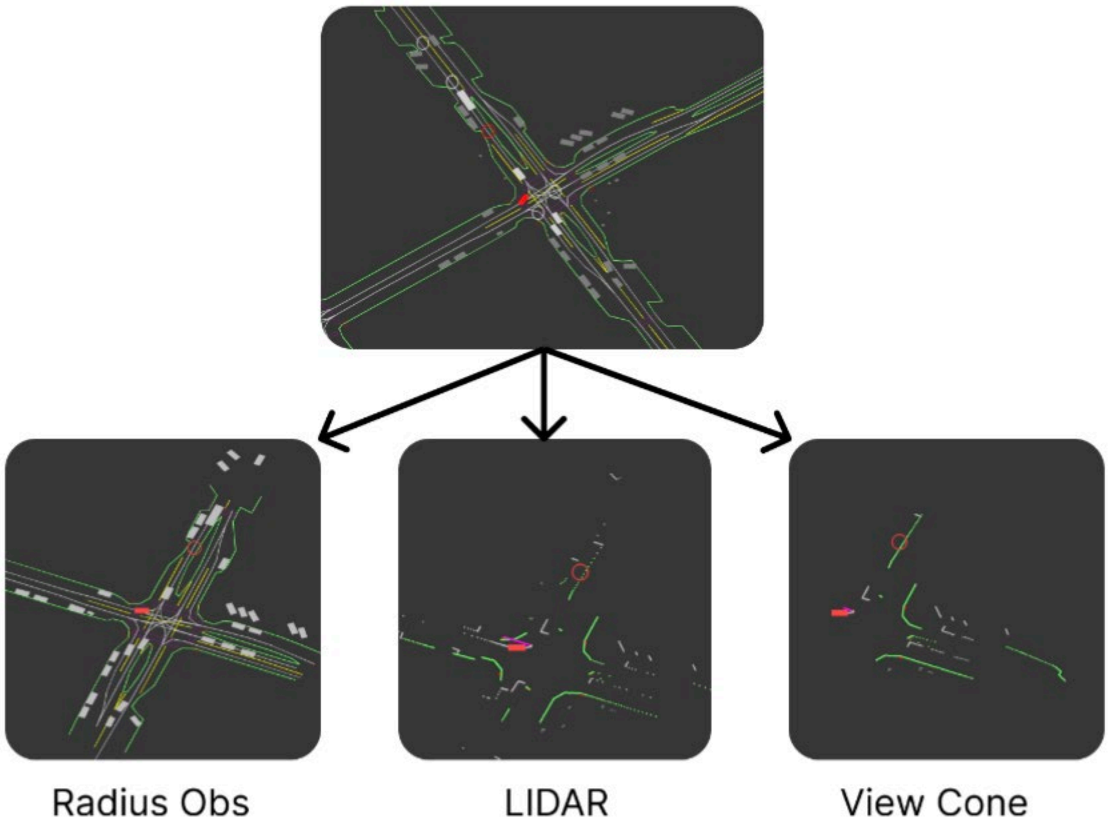


Figure 2: Visualization of different observation spaces available in GPUDrive. The top scene is an example scenario from the dataset, rendered from the ego-centric perspective of the red vehicle. Grey cars are parked cars while white cars are other controlled agent

- **Dataset:** GPUDrive represents its map as a series of polylines and does not require a connectivity map of the lanes. As such, it can be made compatible with most driving datasets given the pre-processing of the roads into the polyline format. Currently, GPUDrive supports the Waymo Open Motion Dataset (WOMD) [60] which is available under a non-commercial license. The WOMD consists of a set of over 100,000 multi-agent traffic scenarios, each of which contains the following key elements: 1) Road map - the layout and structure of a road, such as a highway or parking garage. 2) Expert human driving demonstrations. 3) Road objects, such as stop signs and crosswalks. Figure 1 depicts an example of an intersection traffic scenario as rendered in GPUDrive.

- **Sensor modalities:** GPUDrive supports a variety of observation spaces intended to enable heterogeneous types of agents. Fig. 1 depicts the three types of supported state spaces. The first mode is somewhat unphysical in which all agents and road objects within a fixed radius are observable to the agent. This mode is intended primarily for debugging and quick testing, enabling a user to minimize the amount of partial observability in the environment. The other two modes are based on a GPU-accelerated LIDAR scan, representing what an autonomous vehicle would be able to see and what a human would likely be able to see respectively. Both modes are based on casting LIDAR rays; to model human vision we simply restrict the LIDAR rays to emanate in a smaller, controllable-sized cone that can be rotated through an action corresponding to head rotation. Note that since all objects are represented as bounding boxes of fixed height, the LIDAR observations are over-conservative as humans while LIDAR scans in reality are usually able to see over the hoods of cars.

- **Agent dynamics:** Agents are stepped using a standard Ackermann bicycle model with actions corresponding to steering and acceleration. This model enables the dynamics of objects to be affected by their length, creating different dynamics for small cars vs. trucks. However, this model is not fully invertible which can make it challenging to use this model for imitation learning. To enable full invertibility for imitation learning, we also support the simplified bicycle model, taken from Waymax [1] which is a double-integrator in the position and velocity and updates its yaw as:

$$\theta_{t+1} = \theta + s_t(v_t\Delta t + \frac{1}{2}a_t\Delta t^2)$$

  where θ is the yaw, s is the steering command, v is the velocity, and a is the acceleration at time t respectively. Δt is the timestep. This model is always invertible given an unbounded set of steering and acceleration actions but is independent of the vehicle length. Note that this model does not factor in the length of the car, causing both long and short objects to have identical dynamics. However, computing the expert actions and then using them to mimic the expert trajectory under this model leads to lower tracking error than the default bicycle model.

- **Rewards:** All agents are given a target goal to reach; this goal is selected by taking the last point observed in the vehicle's logged trajectory. A goal is reached when agents are within some configurable distance $\delta$of the goal. By default, agents in GPUDrive receive a reward of 1 for achieving their goal and otherwise receive a reward of 0. There are additional configurable collision penalties or other rewards based on agent-vehicle distances or agent-road distances though these are not used in this work.

- **Termination conditions:** All agents terminate their episode when they reach their goal or collide with another agent. As it is not clear where an agent should go next after it reaches its goal, we simply remove it from the scene afterwards (we note that an alternative might be to generate a new goal for the agent to drive to). Car and cyclist agents are considered to be in collision when they drive through a road edge while pedestrian agents are allowed to cross road edges.

- **Available driving simulation agents:** We use reinforcement learning to train a set of agents that reach their goals 95% of the time on a subset of 500training scenes. While this number is far below the capability of human drivers, these agents are reactive in a distinct fashion from parametrized driver models in other simulators. In particular, many logged-data simulators construct reactivity by having the driver follow along its logged trajectory but decelerate if an agent passes in front of it. In contrast, these agents can maneuver and negotiate without remaining constrained to a logged trajectory. These trained agents are extremely aggressive about reaching their goals and can be used as an out-of-distribution test for proposed driving agents.

- **Simulator sharp-edges:** We note the following limitations of the benchmark:
    - **Absence of a map:** The current version of the simulator does not have a well-defined notion of lanes or a higher-level road map which makes it challenging for algorithmic approaches that require maps. The absence of this feature also makes it challenging to define rewards such as "stay lane-centered."
    - **Convex objects only:** Collision checking relies on the objects being represented as convex objects.
    - **Unsolvable goals:** Due to incorrect in the Waymo dataset, such as an exit to a parking lot being labelled as an impassable road edge, some agent goals (roughly 2%) are unreachable. For these agents, we default them to simply replaying their logged trajectory and do not treat them as agents.
    - **Variance in controllable agents per scenario:** In the majority of scenes, there are approximately 8-10 agents and an average of 50 parked cars. Additionally, the dataset is gathered from the sensors of an autonomous vehicle, leading to some agents having their initial states recorded only after the first time-step of the simulator. These agents are not included, as incorporating them would necessitate "teleporting" them into the scene, potentially leading to unavoidable collisions with agents deviating from their logged trajectories.

Subsection 3.2: Simulator analysis

<u>Subsubsection 3.2.1</u><u>: Simulator speed</u>

Since scenarios contain a variable number of agents, we introduce a metric called Agent Steps Per Second (ASPS) to measure the sample throughput of the simulator. We define the ASPS as the total number of agents across all worlds in a batch that can be fully stepped in a second:

$$ASPS = \frac{S\sum_{k=1}^{N}|A_k|}{\Delta T}$$

where $A_k$is the number of agents in the k-th world, S is the number of steps taken, and ΔT is the number of seconds elapsed. Figure 2 examines the scaling of the simulator as the number of simulated worlds, which represents the amount of parallelism, increases. To measure performance, we sample random batches of scenarios of size equal to the number of worlds, so that every world is a unique scenario with Kagents. On the left-hand side of Figure 2 we compare the performance of GPUDrive to the original Nocturne version [8] (CPU, no parallelism) and a CPU-accelerated version of Nocturne via Pufferlib (16 CPU cores) Empirically, the maximum achievable AFPS of Nocturne is 15,000 (blue dotted line) though we caution that additional speedups may be possible. In contrast, GPUDrive can reach over a million ASPS on a consumer-grade GPU at 512 worlds (average agents per scenario is 60). This performance also surpasses that of Waymax [1] }, a JAX-based simulator, where we could not run more than 32 environments in parallel due to Out of Memory (OOM) issues. Note that GPUDrive exhibits near-linear scaling of ASPS between

32 and 128worlds on a datacenter-grade NVIDIA GPU and between 32 and 256 worlds on a consumer-grade GPU.

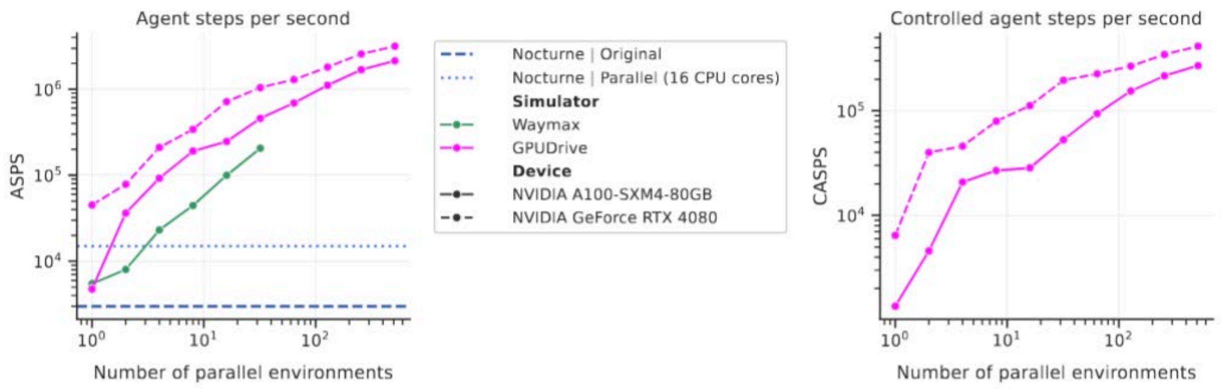


Figure 3: Peak goodput of GPUDrive on a consumer-grade and datacenter-class GPU compared to other GPU and CPU-based implementations. (Left) Total number of agent steps per second. (Right) Total number of controllable agent steps per second. This only accounts for agents that would be controlled in a scene as opposed to parked agents.

<u>**Section 3.2.2:** Training speed</u>

The purpose of GPUDrive is to facilitate research and development in multi-agent algorithms by 1) reducing the completion time of experiments, and 2) enabling academic research labs to achieve scale on a limited computing budget. Ultimately, we are interested in the rate at which a machine learning researcher or practitioner can iterate on ideas using GPUDrive. This section highlights what our simulator enables in this regard by studying the end-to-end process of learning policies in our simulator.

Figure 3 contrasts the number of steps (experience) and the corresponding time required to solve 10 scenarios from the WOMD between Nocturne and GPUDrive. For benchmarking purposes, we say a scene is solved when agents can navigate to their designated target position 95% of the time without colliding or going off-road. Ceteris paribus, GPUDrive achieves a 25-40x training speedup, solving 10 scenarios in less than 15 minutes compared to approximately 10 hours in Nocturne.

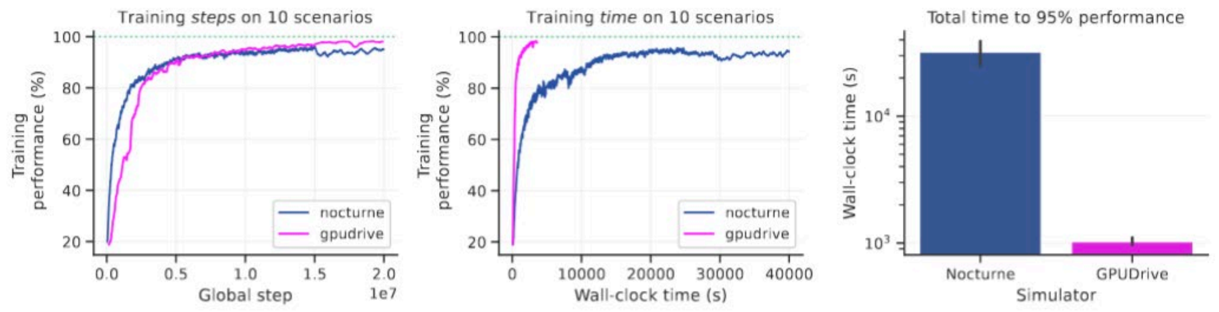


Figure 4: From hours to minutes. *Left:* Training performance (goal-reaching rate) as a function of the global step (AFPS). *Center:* Training performance as a function of wall-clock time. *Right:* Comparison of the total time to solve the same 10 scenarios while replicating environmental and experimental conditions as closely as possible. Runs are averaged across three seeds. The green dotted line indicates optimal performance.

As shown in Figure 3, GPUDrive allows us to solve scenes in minutes. Next, we investigate how the *individual scene completion time*, the time it takes to solve a single scenario, changes as we increase the total number of scenarios we train in. In practice, it may be desirable to train agents on thousands of scenarios. Therefore, we ask whether it is feasible to fully leverage the simulator's capabilities with a single GPU.

Interestingly, we find that the amortized sample efficiency increases with the size dataset of scenes we train in. Figure 4 shows the average completion time per scenario as we increase the dataset. For instance, using IPPO with 32 scenarios takes 2 minutes per scenario. In contrast, solving 1024 unique scenarios takes about 200 minutes, which amounts to only 15 seconds per scenario. We expect that these scaling benefits will continue as we further increase the size of the training dataset. This suggests that GPUDrive should enable effective utilization of the large WOMD dataset comprising 100,000 diverse traffic scenarios, even with limited computational resources.

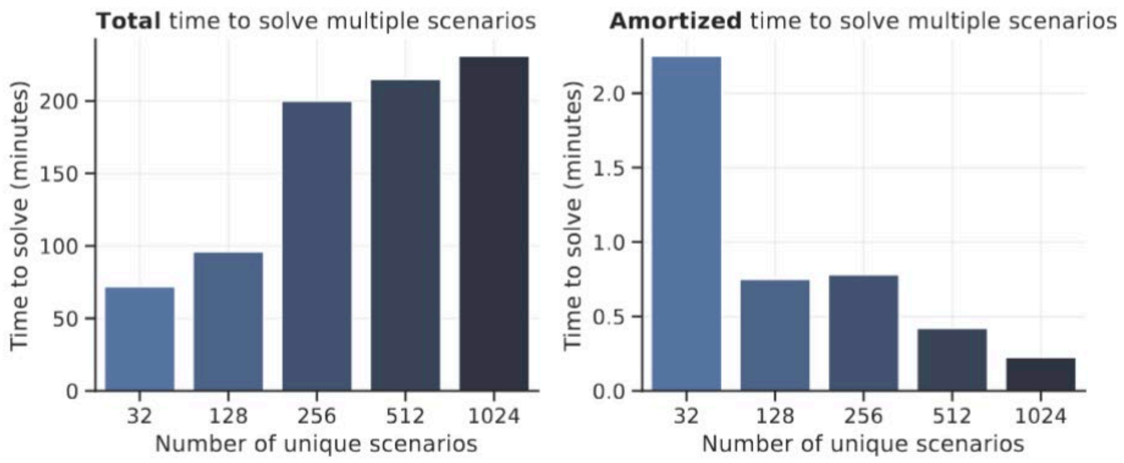


Figure 5: Scale reduces individual scene completion time. ***Left:*** **Total time required to solve a fixed number of scenarios to a goal-reaching rate of 95%. Note that time-to-completion is sub-linear concerning the number of scenes.** ***Right:*** **Each additional scenario costs less to solve than the previous scenario. At 1024 scenes, the per-scene cost of solving an additional scene is on the order of 15 seconds.**

Subsection 3.3: Combining imitation learning and RL

Let o_t, a_t denote the observation and action at time step t and $r(o, a)$ the instantaneous reward for the agent that executes action a in state o. The history up to time $T$is defined as $x_t = (o_1, a_1, \dots, a_{T-1}, o_T)$ (e.g. data collected from a rollout). The basic form of a KL-regularized expected reward objective is defined as:

$$E_\pi = \sum_t \gamma^t r(o_t, a_t) - \lambda D_{KL}(\pi(* \mid o_t) || \pi_{imitation}(* \mid o_t))$$

where π is the most recent **stochastic policy,** $\pi_{imitation}$ is a *stochastic behavioral policy* obtained from a dataset D and λ denotes the *regularization weight*. The KL divergence is defined as the expectation of the logarithmic differences between the pre-trained (fixed) human-policy and RL policy action probability distributions.

Expert demonstrations:
We obtain a dataset of observation-action pairs $D_k = (o_t^i, a_t^i, \dots, o_T^N, a_T^N)$ for N vehicles and $T = 80$ time steps, for a set of K traffic scenarios in the Waymo Open Motion Dataset (WOMD [60] The human driver (``expert") actions (acceleration, steering) are inferred from the positions and velocity of the observed positions using a dynamic bicycle model [1]. As the scenarios are recorded by fusing sensors onboard an autonomous vehicle (AV), the inferred positions of the AV are of higher quality compared to those of surrounding non-AV vehicles, which tend to have more noise. Therefore, we only use the demonstrations from the AV vehicles.

Imitation Learning:

We train a Behavioral Cloning (BC) policy on the shuffled dataset of observation-action pairs to an open-loop accuracy of 97-99%. The dataset, $D_k = (o_t^i, a_t^i, \dots, o_T^N, a_T^N)$ is obtained from $K = 200$ scenarios with $T = 90$ time steps, which is equal to just **30 minutes of driving data**. We obtain the behavioral reference policy $\pi_{imitation}$using the negative log-likelihood objective to the expert demonstrations and implement the algorithm using the *imitation* package [72]. Figure 5 compares the performance of BC policies trained and evaluated on randomly assigned vehicles to only AV vehicles. We also show the performance obtained with the discretized expert actions (top-row), which is an upper bound on performance with this action space. Our BC policy trained on only the AV demonstrations performs better when used to control either the AVs or the random (non-AV) vehicles in the scenarios. Therefore, we select this policy as a regularizer in the multi-agent human-regularized PPO setting.

Agent	Action Space	Generate data from	Evaluate on	Off-road Rate (%)	Collision Rate (%)	Goal Rate (%)
Expert-*actions*	21 × 31	AV only	AV only	9.2	3.3	78.0
BC	21 × 31	AV only	AV only	11.0	4.0	73.1
BC	21 × 31	AV only	Random vehicle	16.0	10.4	51.0
BC	21 × 31	Random vehicle	AV only	17.8	9.0	48.4
BC	21 × 31	Random vehicle	Random vehicle	17.2	7.6	46.2

Figure 6: Performance of different imitation setups.

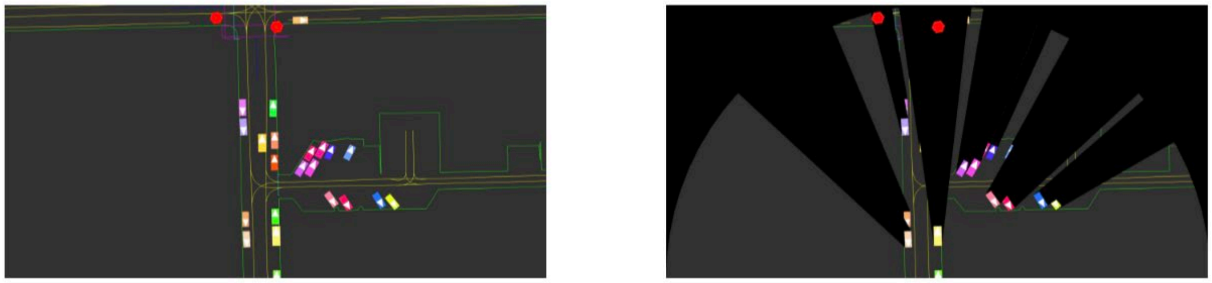


Figure 7: View of an agent. (Left) The whole scene. (Right) what is visible to the green agent at the bottom center of the image.

Subsection 3.4: Evaluating the agents

We use self-play to train HR-PPO agents in scenarios where we control all the vehicles in the scene, with a maximum of 43 controlled vehicles. We compare HR-PPO agents with four different baseline training methods:

- **Multi-agent PPO:** Self-play while controlling all vehicles in the scene, without regularization.
- **Single-agent PPO:** Sample a random agent at reset to control, step the rest of the agents in log-replay.
- **Single-agent HR-PPO:** Add regularization but all but one random agent is in log-replay.
- **Behavioral Cloning:** The behavioral reference policy.

Subsection 3.3.1: Evaluation metrics:

We evaluate our driving agents based on two classes of metrics, as shown in Figure We refer to the first category as *Effectiveness* which measures how well driving agents can achieve their goal safely, without colliding or going off-road. The second category, *Realism*, assesses how closely the driving behavior of the agents matches that of human drivers in the dataset. We use a variation of the Average Displacement Error (ADE) to measure the deviation from the logged human trajectories. In contrast to the trajectory prediction setting, our agents are goal-conditioned and thus they don't have to do inference over their own target goal positions. To distinguish this from the metric used in trajectory prediction, we refer to the metric as the **Goal Conditioned ADE (GC-ADE).** Additionally, we examine the absolute differences between the human expert actions and the policy-predicted steering wheel angle and acceleration at each time step.

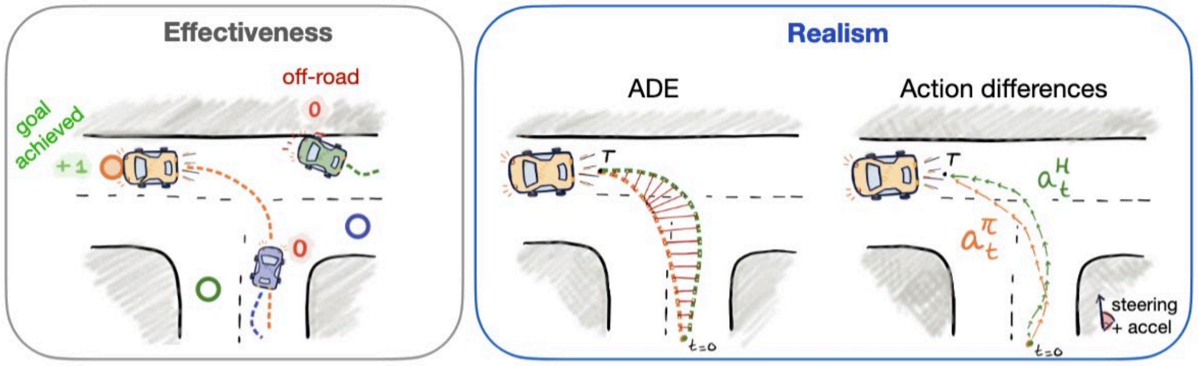


Figure 8 Overview of metrics used for evaluation. Left: Agents achieve their goal if they reach the target (color-coded circles) without collisions before the episode ends (80 steps). In this example, the goal rate is 1/3 (only the yellow car reaches its goal), the off-road rate is 1/3 (the green car hits a road edge) and the collision rate is 0 (no vehicle crashes with another vehicle). Right: Realism metrics concern how agents navigate to their goal positions, that is, the extent to which the policy-generated trajectories (orange) resemble the logged human ones (green).

Findings and Conclusion

Subsection 4.1 – Results of HR:PPO agents

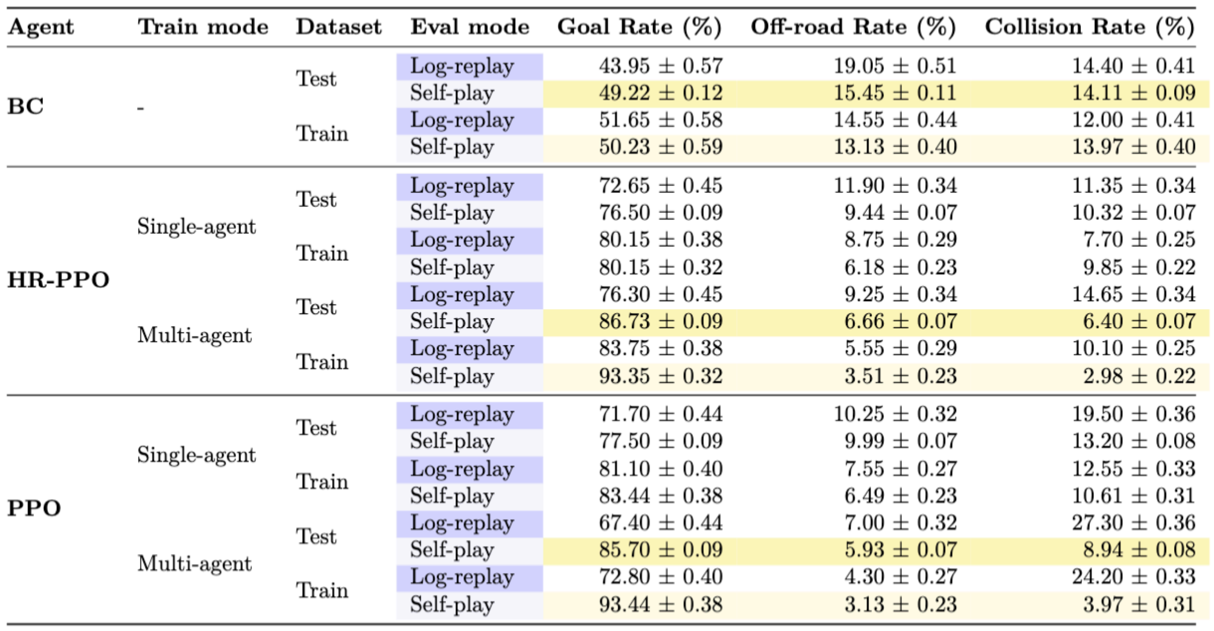
Subsection 4.1.1: Aggregate performance
Figure 9 shows our aggregate performance. We compare the performance of HR-PPO agents to the baselines on the full *train* dataset, which consists of 200 traffic scenarios, and the *test* dataset, consisting of 10000 unseen traffic scenarios. Scenarios have between 1 and 58 vehicles, with an average of 12. We consider two evaluation modes:

- *Self-replay* indicates the setting where we are using a trained policy to control all vehicles in the scenario.
- *Log-replay* indicates that we sample a single, random, vehicle in the scene to control, and the rest of the vehicles are stepped using the **static human replay logs.** To reduce randomness in the performance, we sample each scenario in the dataset 15 times, given that an average of 13 vehicles are included in each scenario. This is distinct from the definition of log-replay in other works [1] where only the AV vehicle (the vehicle used to collect data) is controlled.

We highlight our main findings below.

Agents trained in self-play exhibit the highest performance across all modes:
In closed-loop self-play, the HR-PPO and PPO agents trained in multi-agent mode using self-play achieve the highest performance overall: HR-PPO has a goal rate of 93.35%, an off-road rate of 3.51%, and a collision rate of 2.98%. PPO has a similar goal rate and off-road rate, with a slightly higher collision rate of 3.97%. The standard errors across scenarios are small, typically between 0.5 and 1%. Further, we observe that training in a multi-agent self-play setting is more effective than training in single-agent settings across all test conditions. We find that self-play HR-PPO and PPO agents both outperform their single-agent variants by 10-14%. Surprisingly, even in log-replay evaluation mode, where the self-play agents encounter previously unseen human driving agents, the HR-PPO self-play agents still achieve a 3% improvement over agents **trained directly against the human driving logs**.

Agent-generalization gap decreases using HR-PPO:
Agents trained in self-play typically overfit their training partner. To assess how well the agents can generalize to the unseen human drivers, we compare the change in performance when we switch from *self-play* to *log-replay*. Figure 9 shows that HR-PPO agents have the highest log-replay performance overall and show an improvement of 11% in goal rate and a 14% improvement in collision rate to PPO. Separately, we notice that the train-test gap which combines both agent generalization and scene generalization, is negligible for BC and small for both PPO and HR-PPO, especially given that we train on 200 and evaluate on 10,000 scenes. Overall the performance decreases by approximately 1-8%.



Agent	Train mode	Dataset	Eval mode	Goal Rate (%)	Off-road Rate (%)	Collision Rate (%)
BC	-	Test	Log-replay	43.95 ± 0.57	19.05 ± 0.51	14.40 ± 0.41
			Self-play	49.22 ± 0.12	15.45 ± 0.11	14.11 ± 0.09
		Train	Log-replay	51.65 ± 0.58	14.55 ± 0.44	12.00 ± 0.41
			Self-play	50.23 ± 0.59	13.13 ± 0.40	13.97 ± 0.40
HR-PPO	Single-agent	Test	Log-replay	72.65 ± 0.45	11.90 ± 0.34	11.35 ± 0.34
			Self-play	76.50 ± 0.09	9.44 ± 0.07	10.32 ± 0.07
		Train	Log-replay	80.15 ± 0.38	8.75 ± 0.29	7.70 ± 0.25
			Self-play	80.15 ± 0.32	6.18 ± 0.23	9.85 ± 0.22
	Multi-agent	Test	Log-replay	76.30 ± 0.45	9.25 ± 0.34	14.65 ± 0.34
			Self-play	86.73 ± 0.09	6.66 ± 0.07	6.40 ± 0.07
		Train	Log-replay	83.75 ± 0.38	5.55 ± 0.29	10.10 ± 0.25
			Self-play	93.35 ± 0.32	3.51 ± 0.23	2.98 ± 0.22
PPO	Single-agent	Test	Log-replay	71.70 ± 0.44	10.25 ± 0.32	19.50 ± 0.36
			Self-play	77.50 ± 0.09	9.99 ± 0.07	13.20 ± 0.08
		Train	Log-replay	81.10 ± 0.40	7.55 ± 0.27	12.55 ± 0.33
			Self-play	83.44 ± 0.38	6.49 ± 0.23	10.61 ± 0.31
	Multi-agent	Test	Log-replay	67.40 ± 0.44	7.00 ± 0.32	27.30 ± 0.36
			Self-play	85.70 ± 0.09	5.93 ± 0.07	8.94 ± 0.08
		Train	Log-replay	72.80 ± 0.40	4.30 ± 0.27	24.20 ± 0.33
			Self-play	93.44 ± 0.38	3.13 ± 0.23	3.97 ± 0.31

Figure 9: HR-PPO performance compared to baselines. We report the aggregate mean performance and standard errors across scenarios. *Log-replay* indicates that the agent is evaluated in a single-agent setting where all the other agents are replaying static human driving logs. *Self-play* indicates that all agents in the environment are controlled.

Subsection 4.1.2: Human-likeness:
We aim to construct useful driving agents that can navigate effectively and resemble human driving behavior. To test whether these two properties can be achieved simultaneously, we contrast several existing realism metrics against the effectiveness of agents (Across all four human similarity metrics, we observe that significantly more human-like behavior can be achieved for a minimal or even no trade-off in performance. For instance, Figure 10 shows that HR-PPO with a regularization weight of $\lambda = 0.06$ has a Goal-Conditioned Average Displacement Error (GC-ADE) of 0.54, which is a 60% improvement to PPO (GC-ADE is 1.32), for a decrease in goal rate of 1%, and increase in off-road rate of less than 1%. We observe the same pattern when we compare the policy-predicted actions to the logged human driving logs, as shown in Figure 11, Figure 12, and Figure 13.These measures hold when evaluated in a single-agent setting where we control only the AV vehicles (shown in Figure 14) as well as the setting where we control all vehicles in the scene.

Natural correction for bad actions:
Datasets of human driving may contain noise or undesirable actions. For instance, in our dataset, the off-road rate of replaying the expert actions is quite high (> 12%). However, we observe that HR-PPO agents, which are trained with these imperfect behavioral cloning actions, learn to ignore a large fraction of them and instead achieve an off-road rate between 2-4%. This finding suggests that it may not be necessary to have a near-perfect BC policy as the regularizer as RL can compensate for some of the weaknesses of the regularization policy.

\subsection{Coordinating with human drivers}
\label{sec:coordinating_with_humans}

We explore the ability of HR-PPO agents to coordinate with human drivers in interactive scenarios. Since we cannot directly interact with human drivers, we use the available driving logs as a proxy instead. We compare the collision rates between self-play mode, where all agents are controlled by a single policy, and log-replay mode, where a single random agent is controlled by our policy, and the rest of the agents are controlled by human driving logs. By swapping out only the agents in identical scenarios, we can isolate errors caused by the inability to anticipate other agents' actions.

Figure 15 compares the effectiveness of BC, PPO, and HR-PPO agents in different evaluation modes. PPO performs well when interacting with agents of the same kind but struggles when facing unseen human driver replay agents. Overall, there's a significant increase in collision rates, exceeding 20%, when switching from self-play mode to log-replay mode. HR-PPO also experiences a rise in collision rates, but to a lesser extent, with an increase of 7%. In log replay, HR-PPO outperforms the base BC agent in terms of collision rates while also achieving a much higher goal rate.

The effectiveness of HR-PPO agents in coordinating is more visible when we examine the collision rate as a function of the number of intersecting paths vehicles encounter. Notably, the collision rate for PPO consistently increases as trajectories become more interactive, with collisions occurring between 40-65% of vehicles when encountering one or more intersecting paths. In contrast, the collision rate for HR-PPO shows only a slight increase of approximately 5-8% compared to its self-play collision rate, remaining relatively stable regardless of scene interactivity. It is worth noting that this improvement is not quite evident based on the aggregated performance metrics because more than 70% of all agent trajectories in the dataset do not intersect with other vehicles. Altogether, our results suggest that HR-PPO agents are more compatible with human driving behavior.

What makes HR-PPO agents more compatible with the human logs? To find out, we conduct a qualitative analysis. After analyzing the driving behavior of PPO and HR-PPO agents in 50 randomly sampled scenarios, we conclude that the lower collision rates can be attributed to two main factors. First, the HR-PPO agent's driving style aligns better with human logs, enabling a higher level of anticipation of other agents' actions. Secondly, HR-PPO agents maintain more distance from other vehicles, which reduces the risk of collisions. A subset of videos are available at our site.

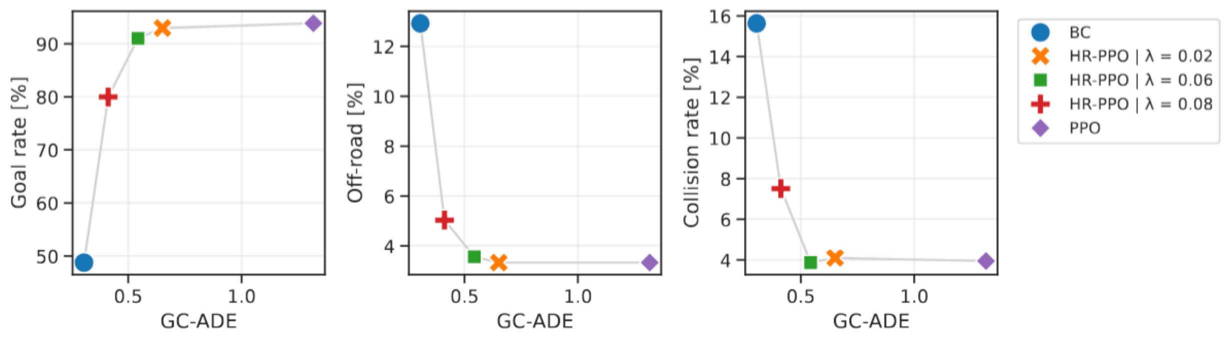


Figure 10: Goal-Conditioned Average Displacement Error (GC-ADE) to logged human driver positions against effectiveness metrics conditioned on knowing the goal. Policies are evaluated on the training dataset of 200 scenarios.

Agent	GC-ADE	Accel MAE	Action Acc. (%)	Speed MAE	Steer MAE
BC	0.31 ± 0.01	1.71 ± 0.02	5.61 ± 0.02	0.84 ± 0.02	0.02 ± 0.00
HR-PPO	0.54 ± 0.01	2.09 ± 0.02	3.25 ± 0.01	1.82 ± 0.03	0.02 ± 0.00
PPO	1.32 ± 0.03	3.93 ± 0.02	0.20 ± 0.00	5.07 ± 0.08	0.08 ± 0.00

Figure 11: Mean and standard error across the 200 scenarios in the training dataset, *controlling all vehicles in every scenario* (Self-play). The reported HR-PPO performance is with $\lambda = 0.06$ (green square in the Figures above).

Agent	GC-ADE	Accel MAE	Action Acc. (%)	Speed MAE	Steer MAE	Goal Rate (%)	Off-Road Rate (%)	Collision Rate (%)
BC	0.08 ± 0.01	0.41 ± 0.02	0.22 ± 0.01	0.09 ± 0.01	0.01 ± 0.00	69.50 ± 1.68	11.00 ± 2.21	6.00 ± 1.68
HR-PPO	0.56 ± 0.03	1.15 ± 0.06	0.10 ± 0.01	1.83 ± 0.08	0.01 ± 0.00	90.00 ± 2.12	1.50 ± 0.86	8.50 ± 1.97
PPO	1.22 ± 0.06	3.92 ± 0.05	0.00 ± 0.00	4.77 ± 0.19	0.09 ± 0.00	71.50 ± 3.19	2.00 ± 0.99	28.00 ± 3.17

Figure 12: Mean performance and standard errors across the training dataset of 200 scenarios, controlling only the AV vehicle in every scenario. This is distinct from the log-replay setting where a random vehicle is set as controlled. The reported HR-PPO performance is with $\lambda = 0.06$.

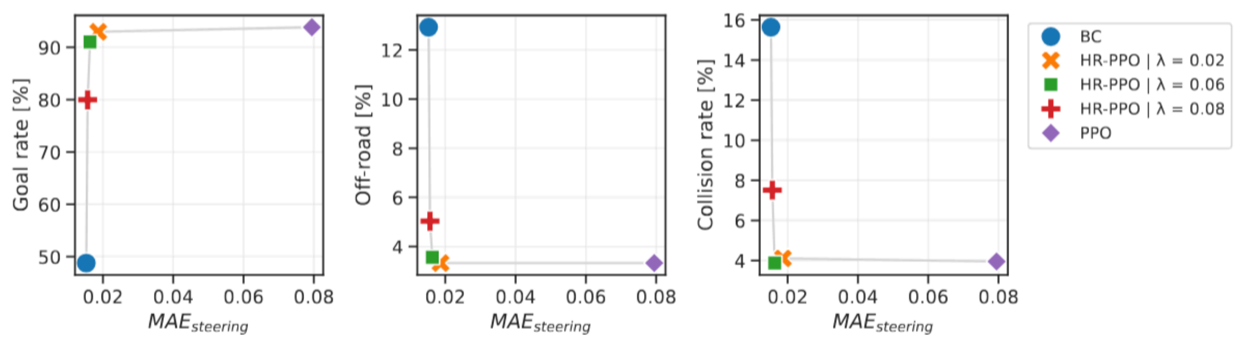


Figure 13: Steering MAE against effectiveness metrics.

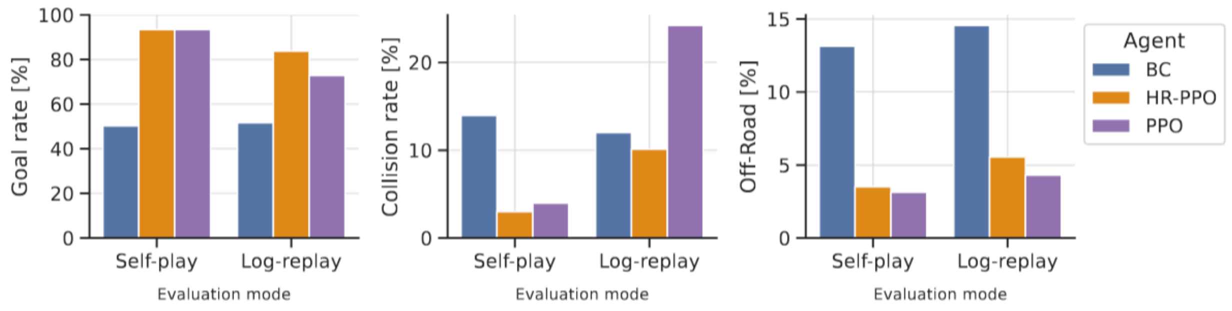


Figure 14: Overall performance gap between evaluating in self-play vs. log-replay settings across the 200 training scenarios.

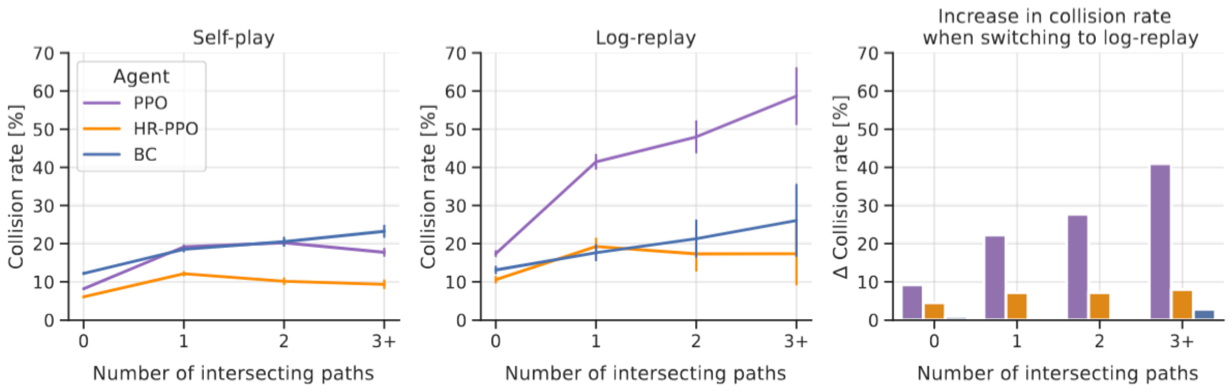


Figure 15: Collision rate change as a function of the number of intersecting paths (a proxy for interactivity)

Subsection 4.2 – Conclusion

In this report we have discussed two new pieces of work: a GPU accelerated simulator that can run at over a million FPS and agents trained in that simulator using imitation regularization. We have demonstrated that the combination shows promising trends in terms of improvements with further training. However, we have not yet achieved our long term goal of training human-level agents in this simulator. In future work, we will remove several training bottlenecks, allowing us to train at the scale of tens of billions of training steps, hopefully leading to sharply improved policies.

References

1. Gulino, Cole, et al. "Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research." *Advances in Neural Information Processing Systems* 36 (2024).
2. Muhammad, Khan, et al. "Deep learning for safe autonomous driving: Current challenges and future directions." *IEEE Transactions on Intelligent Transportation Systems* 22.7 (2020): 4316-4336.
3. Treiber, Martin, Ansgar Hennecke, and Dirk Helbing. "Congested traffic states in empirical observations and microscopic simulations." *Physical review E* 62.2 (2000): 1805.
4. Dosovitskiy, Alexey, et al. "CARLA: An open urban driving simulator." *Conference on robot learning*. PMLR, 2017.
5. Pomerleau, Dean A. "ALVINN: An Autonomous Land Vehicle in a Neural Network." *Advances in Neural Information Processing Systems*, vol. 1, 1988.
6. Xu, Danfei, et al. "BiTS: Bi-Level Imitation for Traffic Simulation." *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 2929-2936.
7. Montali, Nico, et al. "The Waymo Open Sim Agents Challenge." *Advances in Neural Information Processing Systems*, vol. 36, 2024.
8. Vinitsky, Eugene, et al. "Nocturne: A Scalable Driving Benchmark for Bringing Multi-Agent Learning One Step Closer to the Real World." *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 3962-3974.
9. Kreutz, Karsten, and Julian Eggert. "Analysis of the Generalized Intelligent Driver Model (GIDM) for Uncontrolled Intersections." *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2021, pp. 3223-3230.
10. Kesting, Arne, Martin Treiber, and Dirk Helbing. "General Lane-Changing Model MOBIL for Car-Following Models." *Transportation Research Record*, vol. 1999, no. 1, 2007, pp. 86-94. SAGE Publications Sage CA.
11. Caesar, Holger, et al. "NuPlan: A Closed-Loop ML-Based Planning Benchmark for Autonomous Vehicles." *arXiv preprint arXiv:2106.11810*, 2021.
12. López, Pablo Álvarez, et al. "Microscopic Traffic Simulation Using SUMO." *IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2018.
13. Casas, Jordi, et al. "Traffic Simulation with AIMSUN." *Fundamentals of Traffic Simulation*, Springer, 2010, pp. 173-232.

14. Lu, Yiren, et al. "Imitation Is Not Enough: Robustifying Imitation with Reinforcement Learning for Challenging Driving Scenarios." *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2023, pp. 7553-7560.

15. Meta Fundamental AI Research Diplomacy Team (FAIR)†, et al. "Human-Level Play in the Game of Diplomacy by Combining Language Models with Strategic Reasoning." *Science*, vol. 378, no. 6624, 2022, pp. 1067-1074. American Association for the Advancement of Science.

16. Li, Quanyi, et al. "MetaDrive: Composing Diverse Driving Scenarios for Generalizable Reinforcement Learning." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, 2022, pp. 3461-3475.

17. Bojarski, Mariusz, et al. "End to End Learning for Self-Driving Cars." *arXiv preprint arXiv:1604.07316*, 2016.

18. Philion, Jonah, Xue Bin Peng, and Sanja Fidler. "Trajeglish: Learning the Language of Driving Scenarios." *arXiv preprint arXiv:2312.04535*, 2023.

19. Tan, Shuhan, et al. "Language Conditioned Traffic Generation." *arXiv preprint arXiv:2307.07947*, 2023.

20. Ross, Stéphane, Geoffrey Gordon, and Drew Bagnell. "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning." *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, 2011, pp. 627-635.

21. Ng, Andrew Y., and Stuart Russell. "Algorithms for Inverse Reinforcement Learning." *ICML*, vol. 1, 2000, p. 2.

22. Ho, Jonathan, and Stefano Ermon. "Generative Adversarial Imitation Learning." *Advances in Neural Information Processing Systems*, vol. 29, 2016.

23. Fu, Justin, Katie Luo, and Sergey Levine. "Learning Robust Rewards with Adversarial Inverse Reinforcement Learning." *arXiv preprint arXiv:1710.11248*, 2017.

24. Igl, Maximilian, et al. "Symphony: Learning Realistic and Diverse Agents for Autonomous Driving Simulation." *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 2445-2451.

25. Baram, Nir, et al. "End-to-End Differentiable Adversarial Imitation Learning." *International Conference on Machine Learning*, PMLR, 2017, pp. 390-399.

26. Suo, Simon, et al. "TrafficSim: Learning to Simulate Realistic Multi-Agent Behaviors." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10400-10409.

27. Silver, David, et al. "Mastering the Game of Go with Deep Neural Networks and Tree Search." *Nature*, vol. 529, no. 7587, 2016, pp. 484-489.

28. Silver, David, et al. "A General Reinforcement Learning Algorithm That Masters Chess, Shogi, and Go through Self-Play." *Science*, vol. 362, no. 6419, 2018, pp. 1140-1144.

29. Vinyals, Oriol, et al. "Grandmaster Level in StarCraft II Using Multi-Agent Reinforcement Learning." *Nature*, vol. 575, no. 7782, 2019, pp. 350-354.
30. Bard, Nolan, et al. "The Hanabi Challenge: A New Frontier for AI Research." *Artificial Intelligence*, vol. 280, 2020, p. 103216. Elsevier.
31. Strouse, DJ, et al. "Collaborating with Humans without Human Data." *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 14502-14515.
32. Knox, W. Bradley, et al. "Reward (Mis) Design for Autonomous Driving." *Artificial Intelligence*, vol. 316, 2023, p. 103829.
33. Bakhtin, Anton, et al. "No-Press Diplomacy from Scratch." *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 18063-18074.
34. Hu, Hengyuan, et al. "“Other-Play” for Zero-Shot Coordination." *International Conference on Machine Learning*, PMLR, 2020, pp. 4399-4410.
35. Hu, Hengyuan, et al. "Off-Belief Learning." *International Conference on Machine Learning*, PMLR, 2021, pp. 4369-4379.
36. Pan, Xinlei, et al. "Virtual to Real Reinforcement Learning for Autonomous Driving." *arXiv preprint arXiv:1704.03952*, 2017.
37. Liang, Xiaodan, et al. "CIRL: Controllable Imitative Reinforcement Learning for Vision-Based Self-Driving." *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 584-599.
38. Zhang, Chris, et al. "Learning Realistic Traffic Agents in Closed-Loop." *arXiv preprint arXiv:2311.01394*, 2023.
39. Wu, Jingda, et al. "Human-Guided Reinforcement Learning with Sim-to-Real Transfer for Autonomous Navigation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.
40. Hester, Todd, et al. "Deep Q-Learning from Demonstrations." *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.
41. Lerer, Adam, and Alexander Peysakhovich. "Learning Existing Social Conventions via Observationally Augmented Self-Play." *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 2019, pp. 107-114.
42. Hu, Hengyuan, et al. "Human-AI Coordination via Human-Regularized Search and Learning." *arXiv preprint arXiv:2210.05125*, 2022.
43. Jacob, Athul Paul, et al. "Modeling Strong and Human-Like Gameplay with KL-Regularized Search." *International Conference on Machine Learning*, PMLR, 2022, pp. 9695-9728.
44. Meta Fundamental AI Research Diplomacy Team (FAIR)†, et al. "Human-Level Play in the Game of Diplomacy by Combining Language Models with Strategic Reasoning." *Science*, vol. 378, no. 6624, 2022, pp. 1067-1074. American Association for the Advancement of Science.

45. Yu, Chao, et al. "The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games." *Advances in Neural Information Processing Systems*, vol. 35, 2022.
46. Schulman, John, et al. "Proximal Policy Optimization Algorithms." *arXiv preprint* arXiv:1707.06347, 2017. arXiv.org.
47. Bettini, Matteo, Ryan Kortvelesy, Jan Blumenkamp, and Amanda Prorok. "VMAS: A Vectorized Multi-Agent Simulator for Collective Robot Learning." *Distributed Autonomous Robotic Systems*, Springer Proceedings in Advanced Robotics, vol. 28, Springer, 2022, pp. 42–56.
48. Ansel, Jason, et al. "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation." *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, vol. 2, ACM, 2024, pp. 929–947.
49. Bronstein, Eli, et al. "Hierarchical Model-Based Imitation Learning for Planning in Autonomous Driving." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 8652–8659.
50. Packer, Charles, et al. "Is Anyone There? Learning a Planner Contingent on Perceptual Uncertainty." *Proceedings of the Conference on Robot Learning (CoRL)*, vol. 205, PMLR, 2022, pp. 1607–1617.
51. Fisac, Jaime F., et al. "Probabilistically Safe Robot Planning with Confidence-Based Human Predictions." *Proceedings of Robotics: Science and Systems XIV*, 2018.
52. Jaderberg, Max, et al. "Human-Level Performance in First-Person Multiplayer Games with Population-Based Deep Reinforcement Learning." *arXiv preprint* arXiv:1807.01281, 2018. arXiv.org.
53. Silver, David, et al. "Mastering the Game of Go Without Human Knowledge." *Nature*, vol. 550, no. 7676, 2017, pp. 354–359.
54. Pérolat, Julien, et al. "Mastering the Game of Stratego with Model-Free Multiagent Reinforcement Learning." *arXiv preprint* arXiv:2206.15378, 2022. arXiv.org.
55. Wurman, Peter R., et al. "Outracing Champion Gran Turismo Drivers with Deep Reinforcement Learning." *Nature*, vol. 602, no. 7896, 2022, pp. 223–228.
56. Cui, Brandon, et al. "Adversarial Diversity in Hanabi." *Proceedings of the Eleventh International Conference on Learning Representations (ICLR)*, 2023.
57. Zaheer, Manzil, et al. "Deep Sets." *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 3391–3401.
58. Cornelisse, Daphne, and Eugene Vinitsky. "Human-Compatible Driving Partners through Data-Regularized Self-Play Reinforcement Learning." *arXiv preprint* arXiv:2403.19648, 2024.
59. Harris, Charles R., et al. "Array Programming with NumPy." *Nature*, vol. 585, 2020, pp. 357–362.

60. Ettinger, Scott, et al. "Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset." *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, IEEE, 2021, pp. 9690–9699.
61. Lavington, Jonathan Wilder, et al. "TorchDriveEnv: A Reinforcement Learning Benchmark for Autonomous Driving with Reactive, Realistic, and Diverse Non-Playable Characters." *arXiv preprint* arXiv:2405.04491, 2024.
62. Wymann, Bernhard, et al. "TORCS, the Open Racing Car Simulator." 2000. Software available at torcs.sourceforge.net.
63. Leurent, Edouard, et al. "An Environment for Autonomous Driving Decision-Making." 2018.
64. Rajamani, Rajesh. *Vehicle Dynamics and Control*. Springer Science & Business Media, 2011.
65. Chen, Li, et al. "End-to-End Autonomous Driving: Challenges and Frontiers." *arXiv preprint* arXiv:2306.16927, 2023.
66. Bradbury, James, et al. *JAX: Composable Transformations of Python+NumPy Programs*. Version 0.3.13, 2018, github.com/google/jax.
67. Young, Kenny, and Tian Tian. "MinAtar: An Atari-Inspired Testbed for Thorough and Reproducible Reinforcement Learning Experiments." *arXiv preprint* arXiv:1903.03176, 2019.
68. Osband, Ian, et al. "Behaviour Suite for Reinforcement Learning." *arXiv preprint* arXiv:1908.03568, 2019.
69. Bonnet, Clément, et al. "Jumanji: A Diverse Suite of Scalable Reinforcement Learning Environments in JAX." *arXiv preprint* arXiv:2306.09884, 2024.
70. Rutherford, Alexander, et al. "Jaxmarl: Multi-agent RL Environments in JAX." *arXiv preprint*, arXiv:2311.10090, 2023.
71. Shacklett, Brennan, et al. "An Extensible, Data-Oriented Architecture for High-Performance, Many-World Simulation." *ACM Transactions on Graphics (TOG)*, vol. 42, no. 4, 2023, pp. 1–13. ACM, New York, NY, USA.
72. Gleave, Adam, et al. "imitation: Clean imitation learning implementations." *arXiv preprint arXiv:2211.11972* (2022).