Beyond Behavior Cloning in Autonomous Driving: a Survey of Closed-Loop Training Techniques

Peter Karkus*, Maximilian Igl*, Yuxiao Chen, Kashyap Chitta, Jef Packer, Bertrand Douillard, Ran Tian, Alexander Naumann, Guillermo Garcia-Cobo, Shuhan Tan, Alperen Degirmenci, Alexander Popov, Nikolai Smolyanskiy, Urs Muller, Boris Ivanovic, Marco Pavone

***Abstract*—Behavior cloning, the dominant approach for training autonomous vehicle (AV) policies, suffers from a fundamental gap: policies trained open-loop on temporally independent samples must operate in closed-loop where actions influence future observations. This mismatch can cause covariate shift, compounding errors, and poor interactive behavior, among other issues. Closed-loop training mitigates the problem by exposing policies to the consequences of their actions during training. However, the recent shift to end-to-end ("sensor to action") systems has made closed-loop training significantly more complex, requiring costly high-dimensional rendering and managing sim-to-real gaps. This survey presents a comprehensive taxonomy of closed-loop training techniques for end-to-end driving, organized along three axes: action generation (policy rollouts vs. perturbed demonstrations); environment response generation (real-world data collection, AV simulation, generative video and latent world models); and training objectives (closed-loop imitation, reinforcement learning, and their combinations). We analyze key trade-offs along each axis: on-policy vs. on-expert action generation, environment fidelity vs. cost, and expert vs. reward-based training objectives; as well as coupling factors, such as rollout deviation from the policy, expert, and real world logs; and data type, throughput, and latency requirements. The analysis reveals gaps between current research and industry practice, and points to promising directions for future work.**

I. Introduction

Autonomous vehicles (AVs) operate in complex, dynamic environments requiring robust decision-making across diverse driving conditions, from routine lane keeping to rare, high-stakes interactions. To train driving policies, the prevailing paradigm is imitation learning, most commonly in the form of behavior cloning (BC), which frames policy learning as a supervised prediction task using human driving demonstrations. BC is attractive due to its simplicity, scalability, and compatibility with standard supervised learning infrastructure. However, BC has inherent limitations: it trains policies open-loop (OL), assuming independent and identically distributed inputs, while deployment requires closed-loop (CL) operation where inputs, outputs, and future states are interdependent. This OL-CL gap often leads to covariate shift, compounding errors, and causal confusion, and BC alone may also struggle to generalize to long-tail events and dense interactions.

The AV industry has typically addressed BC's limitations by scaling up the volume and diversity of training data, which enabled significant progress in advanced driver-assistance systems. At the same time, BC's scaling behavior remains poorly understood, and it is unclear whether data collection alone can scale sufficiently to achieve higher levels of autonomy [1–3].

The authors are with NVIDIA, Santa Clara, CA, USA. Marco Pavone is also affiliated with Stanford University, CA, USA. *denotes equal contribution.

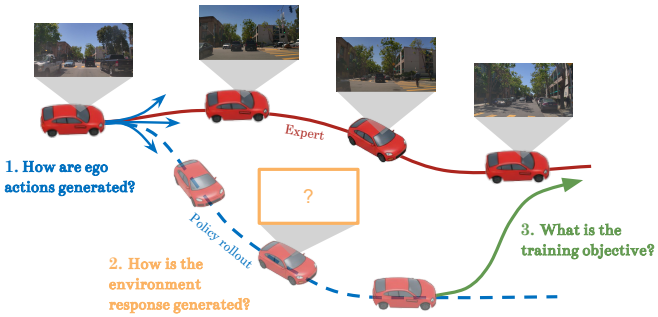


Fig. 1. Closed-loop (CL) training techniques can be examined along three key design dimensions. **1. How ego actions are generated**: actively, by iteratively executing the policy ("rolling it out"); or passively, by augmenting demonstrations. **2. How the environment response is generated**: e.g. 3D grounded simulation, video generation, or latent world models. **3. What the training objective is**: imitation in CL or reinforcement learning.

Alternatively, researchers have explored closed-loop training techniques that expose policies to their own actions' consequences during training, directly addressing the OL-CL gap. Most prior work, however, focuses on policies operating on vectorized environmental representations (e.g., object lists) rather than raw sensory inputs (e.g., images) [4]. This approach has worked well when AV systems were modular, combining perception, prediction, and planning through carefully engineered stacks [5–8].

Recent advances in machine learning, vision-language-action models, and massive driving datasets have pushed the AV industry toward end-to-end learned systems, where the policy maps raw or processed sensor inputs directly to trajectories or control actions [9–13]. Closed-loop training for these policies presents greater complexity: high-dimensional observations, costly rendering, and increased sim-to-real gaps from complex input spaces. These challenges force practitioners to navigate difficult trade-offs and design choices.

Contribution. This paper comprehensively reviews CL training techniques for end-to-end driving policies that extend beyond pure BC. Unlike prior surveys that focus on individual techniques like reinforcement learning, we broadly address the OL-CL gap, covering techniques from CL augmentations to imitation learning to full reinforcement learning. We examine CL training techniques through three orthogonal design axes illustrated in Fig. 1: how actions are generated, how

environmental responses are generated, and what the training objective is. This structure yields a *taxonomy* of prior works, shown in Fig. 4. The taxonomy gives rise to a structured literature review and an analytical framework. We analyze key trade-offs along each axis (on-policy vs. on-expert action generation, environment fidelity vs. cost, expert vs. reward-based objectives) as well as coupling factors (rollout deviation, data requirements); and review the literature in a structured manner, highlighting unique challenges and identifying open research directions.

The paper is structured as follows. Section II provides background and problem formulation. Section III discusses limitations of OL training and OL-only mitigation strategies. Section IV provides a taxonomy of CL training techniques and key design trade-offs. The following sections then discuss action generation in Section V, environment responses in Section VI, and training objectives in Section VII. Section VIII provides conclusions.

II. Background

Problem formulation. This survey focuses on *end-to-end driving policies*, that is, learned functions that directly map a history of sensory observations to the driving actions of the autonomous vehicle ("ego"). The task can be expressed as a *Partially Observable Markov Decision Process*, $(\mathcal{S}, \mathcal{O}, \mathcal{A}, T, Z, R)$, where $s_t \in \mathcal{S}$ denotes the hidden world state at time t, $o_t \in \mathcal{O}$ the raw sensory observations, which may include (a history of) images from multiple cameras, lidar or radar data, ego-motion estimates, HD maps, and user inputs such as route or desired speed, and $a_t \in \mathcal{A}$ the control actions, typically continuous throttle, brake, and steering signals. $T(s_{t+1} \mid s_t, a_t)$ defines environment dynamics, $Z(o_t \mid s_t)$ the observation model, and $R(s_t, a_t)$ the reward. These functions are generally unknown in the real world, so learning has to rely on demonstrations or surrogate objectives.

An end-to-end policy is represented as $\overline{a}_t \sim \pi_\theta(\overline{a} \mid o_t)$, where θ denotes learned parameters, o_t the input observation, and $\overline{a}_t$ an abstract output action. The abstract action may correspond to low-level control commands or, more commonly, a short-horizon (1–8s) trajectory tracked by a downstream controller. Without loss of generality, its relation to raw controls is written as $a_t = f_c(\overline{a}_t)$. Policy architectures may instantiate π_θ differently: deterministic models perform direct regression, probabilistic models output parameterized distributions, recurrent models incorporate memory, and generative models such as tokenized or diffusion-based ones produce actions through iterative decoding. We assume a probabilistic model and highlight cases where the discussion depends on architectural choices.

In the context of policy learning, for brevity, we abuse terminology and refer to the observation o_t and abstract action $\overline{a}_t$ simply as *state* and *action*, and use a_t in place of $\overline{a}_t$ in notation. We further define a generalized dynamics function $f_d(o_t \mid o_{t-1}, a_{t-1}) = Z(o_t \mid s_t)\, T(s_t \mid s_{t-1}, f_c(a_{t-1}))$ that captures the temporal relationship between policy outputs and subsequent inputs, encompassing the effects of the controller f_c, environment dynamics T, and observation model Z.

Open-loop vs. closed-loop training. Driving is inherently a closed-loop process: the policy operates sequentially, and each output a_t affects future inputs $o_{t:T}$ through the dynamics f_d. *Open-loop (OL)* training, such as BC, does not explicitly account for this temporal feedback, and treats policy learning as a supervised prediction task on an independent and identically distributed (IID) dataset of expert demonstrations, $\mathcal{D} = \{(o_t^E, a_t^E)\}$, collected under an expert policy π_E.

Closed-loop (CL) training, in contrast, explicitly accounts for temporal feedback by *rolling out* the policy during training. A rollout $r = (o_1, a_1, \ldots, o_T, a_T)$ is generated according to the policy and environment as

$$p_\theta(r) = \prod_{t=1}^{T} \pi_\theta(a_t \mid o_t)\, f_d(o_t \mid o_{t-1}, a_{t-1}).$$

In practice, CL training often employs *action generation* strategies that sample from a modified policy $\pi'_\theta \sim \pi_\theta$ with added exploration, expert guidance, or other augmentations. Since generating the *environment responses* via f_d is costly in the real-world, CL training typically relies on simulators, learned models, or other surrogate environments. The *training objective* then evaluates performance along rollouts, e.g., by maximizing cumulative reward as in reinforcement learning, or by aligning induced behavior with expert demonstrations as in CL imitation learning.

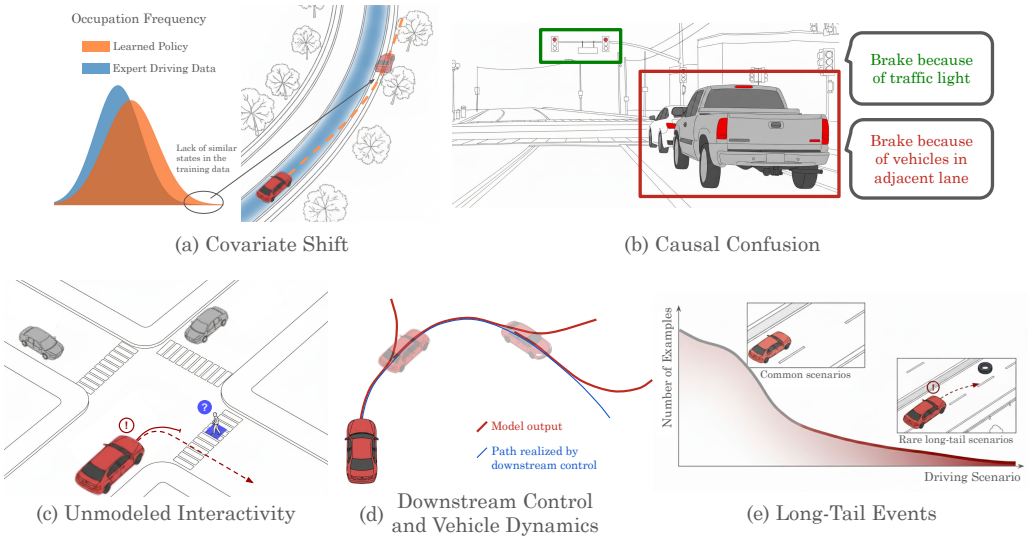
Relation to prior surveys. Reflecting the considerable research interest in autonomous driving, numerous surveys have reviewed progress in AV technologies [9, 14, 15], addressing subproblems such as perception [16], motion prediction [17], and planning and control [18]; as well as the adoption of specific technologies, including deep learning [19, 20], world models [21], and foundation models [22].

Most relevant to our work are recent surveys [4, 23, 24] on (open-loop) imitation and reinforcement learning for AVs. We review the literature from a distinct angle: on one hand, focusing on reducing the OL-CL gap in a broader sense, covering a wider range of techniques including CL imitation; and from the perspective of synthetic sensor data generation, which gains utmost importance in the context of end-to-end driving policies, as opposed to vectorized policies that were the focus of most prior surveys. Our work is timely: in recent years, end-to-end policies have become prevalent in the AV industry, while advances in visual simulation, such as neural rendering [25], video generative models [26], and latent world models [27], offer new opportunities for closed-loop training.

Related problems. Several distinct problems in the literature share design questions and techniques with end-to-end driving policies, such as:

- *Modular AV planning:* Vectorized policy learning and planning (vectorized rather than raw sensory inputs).
- *Traffic simulation:* Creating policies for simulated traffic agents.
- *Motion forecasting:* Predicting other road users' behavior.
- *Robot learning:* Learning policies for robot navigation or manipulation.

While not the main focus of this survey, we reference work from these fields where relevant.



(a) Covariate Shift (b) Causal Confusion

(c) Unmodeled Interactivity (d) Downstream Control and Vehicle Dynamics (e) Long-Tail Events

Fig. 2. **The OL-CL gap.** Training a policy in open-loop (OL) and deploying it in closed-loop (CL) creates an OL-CL gap, presenting key challenges: a) covariate shift, b) causal confusion, c) unmodeled interactivity, d) downstream control and vehicle dynamics, and e) long-tail events.

III. Limitations of Open-Loop Training

A. The open-loop/closed-loop gap

Open-loop training techniques, like BC, train the policy as an IID prediction task, mapping observations to expert actions, without temporal feedback. During deployment, however, policies operate sequentially in *closed-loop*, where each decision influences future observations, thus the IID assumption no longer holds. More broadly, the OL–CL mismatch leads to lack of robustness from covariate shift (compounding errors), causal confusion, unmodeled effects from downstream control and interactions, and poor performance in long-tail events. Beyond theoretical results [28, 29], recent AV research also highlights the practical impact of the OL-CL gap [30–32]. We illustrate the OL-CL gap in Fig. 2, and review its components in detail next.

a) Covariate shift. Covariate shift (also called distribution shift), arguably the most important issue, occurs when the state distribution encountered by the deployed policy deviates from that in the training data [28]. Since expert demonstrations for BC cover only a subset of possible states, the policy may perform well in familiar conditions yet fail catastrophically in unseen ones. As the policy is rolled out in CL, small deviations from the expert amplify over time (compounding errors), pushing the policy into less and less familiar regions of the state space (e.g., driving too far off the centerline), further degrading the policy until it can no longer recover.

b) Causal confusion. Causal confusion arises when the policy relies on spurious correlations in the training data, yielding high OL accuracy but failing in CL [33]. A classic example is the *copycat agent* [34], which predicts actions primarily from its own past actions or other nuisance variables, rather than from environment observations. Similarly, at a red light a BC policy may learn to slow down when surrounding vehicles slow down, rather than when the traffic light is red.

c) Unmodeled interactivity. In autonomous driving, the reactions of other road users create feedback loops: the ego vehicle's behavior influences others, which in turn alters the ego vehicle's future observations. BC, trained solely on the passive observation of expert demonstrations, does not model this bidirectional influence. Consequently, the policy may learn behaviors that appear reasonable in OL but provoke undesirable reactions from other agents in CL.

Further, the practice of predicting fixed-horizon trajectories, rather than instantaneous actions or reactive policies, is ill-posed [35]. For example, a stopped AV cannot know the exact timing of a future traffic light change, yet BC forces it to commit to a specific plan. Similarly, behaviors like yielding vs. nudging unfold over multi-stage interactions, but BC captures them only as multimodal trajectory predictions, a non-causal proxy for an extended interactive process.

d) Downstream control and vehicle dynamics. Predicted trajectories are not executed perfectly: they must obey vehicle dynamics and are typically passed through a downstream controller (e.g., PID, MPC), which introduces latency, noise, and actuation limits. Since BC policies are not exposed to downstream control and vehicle dynamics during training, they can fail to appropriately account for them.

e) Long-tail events. Long-tail events, i.e., rare but critical, out-of-distribution (OOD) scenarios, pose a persistent challenge for AVs [36, 37]. BC is vulnerable because rare events are underrepresented in expert data, and they contribute little to the training loss. In part due to the reliance on expert demonstrations, expanding the training data to rare events is costly; and without corrective feedback during training the policy often cannot adapt to or recover from rare situations.

B. Mitigation strategies within the open-loop training paradigm

There are a number of popular strategies that can partially mitigate the OL-CL gap without requiring CL training.

First, expert demonstrations can be **reweighted or resampled**, favoring harder and more interesting examples, ultimately to better align the distribution of states during training

to states encountered during CL deployment [38]. Data augmentation techniques, such as random cropping, flipping, color jittering, or injecting other types of noise, can also be applied to increase the diversity of training data [39–43]. Second, policies are often trained to **predict trajectories longer than required** by the downstream controller, encouraging the model to account for longer-term consequences that will arise in CL rollouts [24]. The imitation loss is also often augmented with other **trajectory-level costs**, e.g., for collisions, lane departures, or other safety violations [44]. These costs are akin to offline RL (see Section VII-B), and act as proxies for exposing the policy to the consequences of its actions in CL deployment. Finally, the limitations of OL training are often addressed by simply **scaling up expert demonstration data** using large fleets of real world vehicles. However, the scaling laws for autonomy are not fully understood [2]: e.g., improving from 99% to 99.9% reliability may require anywhere from $10\times$ to $10{,}000\times$ more data. Collecting such volumes is costly and still leaves gaps for rare or safety-critical scenarios.

Despite notable progress toward fully autonomous vehicles, it remains uncertain whether OL training alone can sufficiently close the OL–CL gap, and whether doing so will be practical.

IV. Taxonomy of Closed-Loop Training Approaches

To overcome the limitations of OL training, CL training exposes the policy to the consequences of its own actions during training. At a high level, the recipe is simple: generate additional relevant experience that (i) results from iteratively executing the policy ("rolling it out") and (ii) is realistic enough to avoid sim-to-real failure modes, then (iii) update the policy with an objective aligned to desirable driving performance. However, in practice it is not feasible to achieve all three perfectly and resulting approximations induce fundamental tradeoffs that motivate the analysis in this paper.

Taxonomy. A central contribution of this survey is a taxonomy of CL training approaches. The taxonomy is shown in Fig. 4. It organizes the design space of CL training methods along three main axes (illustrated in Fig. 1):

1) **How are ego actions generated?** To expose the policy to new states arising in CL execution, ego actions may be generated *actively* by unrolling the policy itself, or *passively* by augmenting expert trajectories (perturbing states or actions) independently of the learned policy.
2) **How is the environment response generated?** Given new ego actions, the resulting sensor observations need to be generated. They may come from real-world data collection or from simulation that renders camera views, models traffic agent behavior, executes controllers and vehicle dynamics. Recently, video generative models and latent world models have also emerged as alternatives to traditional 3D-grounded simulation.
3) **How is the training objective defined?** Given the generated CL experience, the policy is updated using either an imitation objective derived from expert demonstrations, reinforcement learning objective derived from a reward function, or a combination of the two.

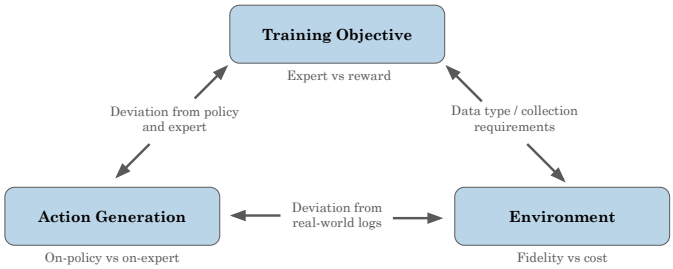


Fig. 3. Design trade-offs coupled between training objective, action generation, and environment response generation.

Design trade-offs. The three axes of CL training are deeply interdependent and practical approaches must carefully balance a set of design trade-offs (illustrated in Fig. 3):

1) **On-policy vs. on-expert action generation.** *On-policy* actions are sampled from the policy under training; *on-expert* actions are sampled close to expert demonstrations. The two are in tension: on-policy actions improve stability but may deviate far from the expert and require costly data collection after every update. On-expert actions enable supervision from demonstrations and reconstruction-based simulation, but being *off-policy*, they do not directly address covariate shift.
2) **Environment fidelity vs. cost.** A high-fidelity environment, in terms of visual appearance, dynamics, and agent interactions, is important to avoid overfitting to artifacts and reduce the sim-to-real gap. Yet they typically come at higher computational or operational cost, limiting throughput and scalability.
3) **Expert vs. reward based training objective.** Expert-driven objectives, as in imitation learning, generally yield more human-like driving than reward-based RL, since specifying comprehensive and robust reward functions is notoriously difficult. However, reliance on experts limits supervision in states far from the demonstration distribution.

Crucially, these axes are coupled, a choice along one axis influences the choices along the others (see arrows in Fig. 3):

4) **Deviation from expert demonstrations.** The choice of training objective constrains action generation, e.g., objectives derived from pre-collected expert trajectories remain reliable only near the expert trajectory, restricting the use of fully on-policy action generation.
5) **Deviation from real-world logs.** The environment choice also constrains action generation. To achieve high fidelity, simulation environments often condition on real-world data, such as with neural reconstruction. However, fidelity typically deteriorates for states far from recorded trajectories, requiring off-policy action generation that remains close to the real-world logs.
6) **Data type / collection requirements.** Different objectives impose distinct data requirements that shape environment choice: e.g.,structured labels (maps, object tracks, etc.) for reward-based objectives; expert labels for imitation objectives; and high throughput, low latency for RL optimization.

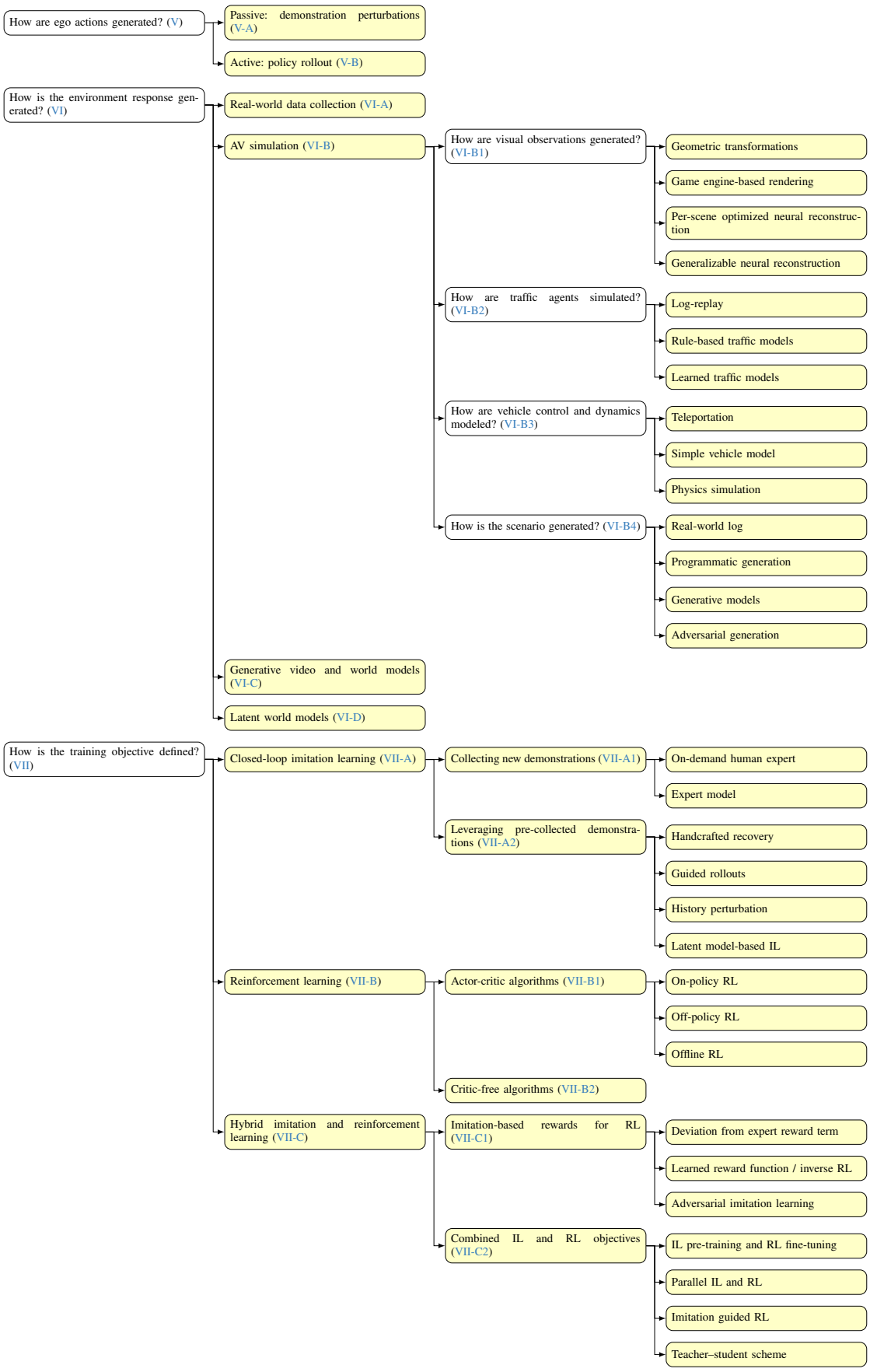


Fig. 4. Taxonomy of closed-loop training methods for autonomous driving, organized along three axes: (1) how ego actions are generated; (2) how the environment response is generated; and (3) how the training objective is defined. White denotes orthogonal design axes, yellow denotes alternative approaches.

In the following sections we review prior work along each axis, highlighting empirical trade-offs and open challenges.

V. How are Ego Actions Generated?

The first step in addressing the OL-CL gap (Section III) is generating ego actions that expose the policy to states it will encounter during CL deployment. While on-policy data most effectively addresses this mismatch, constraints such as computational cost, environment fidelity and the effectiveness of the training objective require careful trade-offs in action generation strategies. Most importantly, on-policy rollouts lead to states far from real-world expert demonstrations, which in turn makes imitation-based objectives less reliable and impacts the fidelity of real-world anchored simulation.

Action generation approaches, therefore, balance between staying close to expert demonstrations and staying close to the policy, as illustrated in Fig. 5. Along this spectrum we categorize action generation approaches into *passive* methods that perturb expert demonstrations independently of the learned policy, and *active* methods that generate states through closed-loop policy rollouts.

A. Passive: demonstration perturbation

Passive perturbations augment expert demonstrations through policy-independent data generation, enabling reuse across training runs and straightforward parallelization. Unlike sensor-level augmentations (see Section III-B), these methods perturb the ego vehicle's state, i.e. position, orientation, or trajectory, necessitating re-rendering of sensor observations from the modified viewpoints (see Section VI). While computationally efficient, they face two key limitations: missing policy-specific failures and, when perturbations are excessive, generating states far out-of-distribution, both of which reduce their effectiveness at addressing the OL-CL gap (Fig. 6).

Two primary approaches exist for state perturbation. **Direct state modification** adds noise directly to the ego vehicle's position and orientation, creating instantaneous deviations from the expert trajectory [44]. While computationally efficient, this approach may produce kinematically implausible states that are problematic for history-dependent policies. **Action perturbation** instead modifies the expert's actions before rolling them out through vehicle dynamics, ensuring kinematic feasibility at the cost of additional computation. When the original data contains only ego locations, this requires first inferring the ground-truth actions, introducing additional uncertainty. Action perturbations are typically implemented by adding noise to the actions themselves or by inducing intentional errors in the tokenization process for transformer-based architectures [45–48].

B. Active: policy rollout

Active rollouts directly address covariate shift by exposing the policy to the consequences of its own actions, revealing compounding errors while providing evaluation signals. However, they require data regeneration after each policy update, incurring significant computational costs. Since policies typically output distributions rather than single actions, several rollout strategies exist. The simplest approach, **on-policy rollout**, randomly samples from the action distribution [32], though this can cause significant deviation from ground truth and produce low-quality rollouts, particularly early in training.

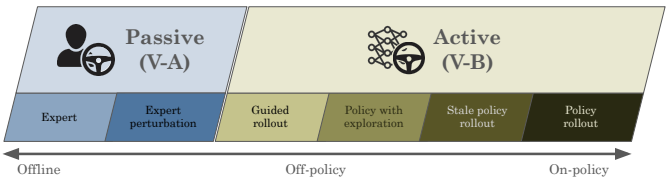


Fig. 5. Action generation approaches in terms of *on-policiness*, i.e. how closely actions follow the current policy.

Several modifications exist to address these limitations. Using **stale policy rollouts** reduces computational costs by reusing data from previous policy versions[1], trading efficiency for reduced effectiveness in addressing the OL-CL gap [49–51]. This particularly impacts RL, requiring algorithmic modifications for stability (e.g. [52, 53]). **Guided rollouts** constrain trajectories to remain near ground-truth paths or trajectories[2], as required by certain rendering technologies and training objectives (Sections VI and VII) [45]. **Quality enhancement** through tree search [35, 54, 55] or other forms of forward search [8, 56] and post-hoc filtering [40] removes undesirable behaviors from the generated data, which is crucial for imitation learning. The filtering process can be viewed as a "policy improvement operator" that improves the policy through data refinement, as in [57]. In contrast, RL requires such negative examples to learn from mistakes. Some RL algorithms even deploy an additional noisy **exploration policy** to encourage exploration and prevent premature convergence [58, 59]. Note that all these modifications result in off-policy rollouts, requiring appropriate adjustments for RL algorithms (see Section VII-B).

The choice of strategy depends on available resources and training objectives: offline methods favor passive perturbations, online imitation learning employs guided rollouts near expert trajectories, while reinforcement learning requires on- or off-policy rollouts, optionally with explicit exploration. Next, we discuss how to generate sensor renderings (Section VI) and supervision signals (Section VII) from these perturbed states.

VI. How is the Environment Response Generated?

The environment generating observations resulting from ego actions, $f_d(o_t \mid o_{t-1}, a_{t-1})$, is a critical component of CL training, especially so for end-to-end policies that take sensor observations as input. Approaches for generating sensor input range from real world drives to learned latent world models. The prominent approach is AV simulation with geometric grounding that renders visual observations using game engines

[1]Stale policy rollouts are often called **off-policy** rollouts in the RL literature. We use "off-policy" more broadly to refer to all rollouts that do not follow the current policy exactly, including stale rollouts, guided rollouts or rollouts using an exploration policy.

[2]Paths denote spatial information; trajectories include timing.

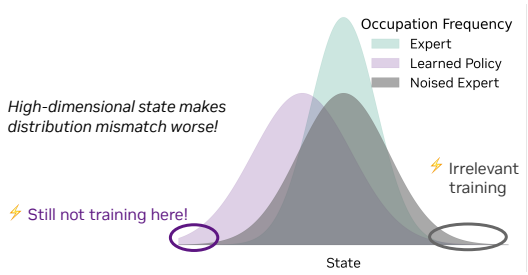


Fig. 6. Passive augmentation of expert demonstrations is computationally efficient but has limited coverage: it fails to capture policy-specific failure modes while potentially generating irrelevant out-of-distribution states.

or neural reconstruction, and independently models traffic agent behavior and vehicle control and dynamics.

The choice of environment is intrinsically linked to both action generation (Section V) and training objectives (Section VII). This relationship involves three fundamental trade-offs as follows.

a) Fidelity vs execution cost. High-fidelity environments incur substantial computational costs, requiring careful trade-offs. Practitioners must choose between: (a) combining data-intensive algorithms like on-policy RL with lower-fidelity environments, or (b) leveraging data-efficient methods such as off-policy RL, closed-loop augmented IL, and offline RL/IL with higher-fidelity but costlier simulations. Additionally, for on-policy RL specifically, simulation *latency* becomes as critical as *throughput*, since each gradient step must wait for newly generated data (see Fig. 8).

b) Fidelity vs generalization ability. The fidelity-generalization trade-off presents a different challenge: simulation technologies anchored to real-world data (e.g., neural reconstruction) achieve superior fidelity but struggle to generalize when trajectories deviate significantly from recorded expert data (see Fig. 7), remaining confined to states near the original data distribution. This limitation necessitates specialized rollout generation techniques, such as guided rollouts or post-hoc filtering (Section V-B), to maintain the ego vehicle within the valid state space.

c) Diversity vs label availability. Finally, the diversity-label availability trade-off reflects the tension between generating sufficiently varied long-tail scenarios and retaining the structure needed for supervision. Capturing long-tail events requires generating or augmenting diverse simulation instances, for example through generative video models or other data-driven approaches. However, such methods often lack grounding in structured scene representations, which are typically necessary to compute rewards in RL-based training or to align with expert demonstrations in IL-based training.

In the following subsections, we examine different environment types used for closed-loop training, ranging from real-world vehicle deployments to various simulation approaches. We organize these by increasing levels of abstraction: direct vehicle data collection (Section VI-A), traditional AV simulation (Section VI-B), generative video models (Section VI-C), and latent world models (VI-D). Each approach presents distinct trade-offs in fidelity, cost, generalization, diversity, and label availability.

A. Real-world data collection

A straightforward approach is to collect onboard data directly from real-world AV deployments. This yields the most realistic CL training data, entirely eliminating the gap between training and deployment. The main disadvantages are the substantial required infrastructure and high costs. Additionally, the generated data has limited utility for certain training algorithms due to high collection latency and premature safety interventions that prevent policies from observing the outcomes of risky actions. Using real world *disengagements* for CL finetuning, i.e., events where the driver takes control from the AV, is a conceptually simple approach, yet it remains largely underexplored in the literature [60].

B. AV simulation

Compared to the real world, simulation enables faster and more cost-efficient data generation. End-to-end AV simulators with geometric grounding build upon a structured, object-centric 3D representation (e.g., roads, vehicles, pedestrians) and then render visual observations from novel viewpoints. The simulated scenes are either based on real world logs, designed by hand, or generated. AV simulation involves several orthogonal design choices, each with distinct trade-offs and challenges: (i) how visual observations are generated (e.g., game-engine rendering, per-scene optimized or generalizable neural rendering); (ii) how traffic-agent behavior is modeled (replayed, handcrafted, or learned); (iii) how vehicle control and dynamics are represented (from simple teleportation to full physics simulation); and (iv) how scenarios are generated (from real-world data to adversarial generation).

Closely related to end-to-end simulators are vectorized AV simulators, which operate directly on structured features (e.g., 2D bounding boxes, lane lines) without generating visual observations. A large portion of CL training and evaluation research relies on vectorized simulators, such as NuPlan [61], Waymax [62], and GPUDrive [63]. Since they cannot be used to train end-to-end policies, they fall outside the scope of this survey. Nevertheless, the design considerations are largely similar, except for rendering, which introduces substantial challenges in terms of sim-to-real transfer and rendering throughput.

1) How are visual observations generated?: **Geometric transformations.** A simple simulation approach is actuated replay, where the GT inputs are transformed analytically to new states encountered during policy rollout. For visual inputs, prior work explored using analytic geometric camera transformations, assuming the states are not far from the GT [38, 64]. Due to incomplete information and assumptions made about the geometry of the world, actuated replay methods are generally limited to small deviations from the original viewpoint, as larger shifts often introduce severe visual artifacts and distortions, especially for dynamic actors. Notably, RAP [65] proposes simple 3D rasterization to generate visual observations, which enables larger viewpoint changes but introduces a significant sim-to-real gap.

Game engine-based rendering. Game engine-based simulators, such as CARLA [66], leverage computer graphics to

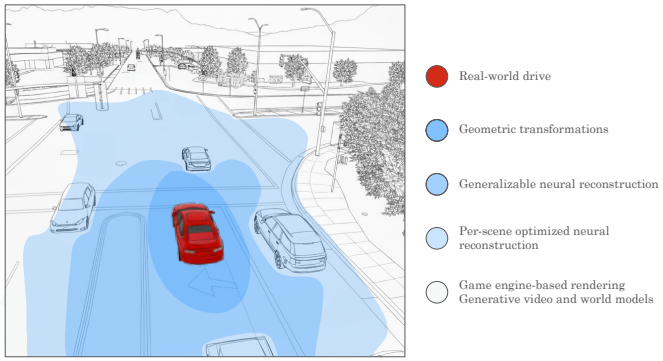


Fig. 7. **Rendering validity corridor**: distance from real-world logs where the environment can produce sufficiently high-fidelity observations. The equivalent behavioral validity corridor is not shown.

render camera views given 3D assets that model the 3D scene, including static and moving parts. Other examples include DriveSim [67] and Metadrive [68], and there is also work on directly using video games such as GTA [69] and TORCS [70]. Despite significant improvements in recent years, perceptual differences between rendered and real-world images remains a challenge. Importantly, game engines require a large number of 3D assets to reconstruct the world, making them difficult to scale, and without anchoring the scene to real-world driving logs, human driving demonstrations are not directly available, which makes applying IL approaches difficult. Accordingly, 3D asset generation and conditioning on real world scenes are important areas of future research [71–73].

Per-scene optimized neural reconstruction. Neural scene representations, such as NeRF and 3DGS, have emerged as a data-driven way to reconstruct 3D scenes, including driving scenes with dynamic objects [74–78], and have enabled reconstruction-based CL simulation, such as AlpaSim [79] and others [80, 81]. Given a set of images, they train a neural representation for the scene, and then render images from novel viewpoints using differentiable volumetric rendering. Unlike game engines, neural reconstruction does not require handcrafted 3D assets and is directly anchored to real world drives. At the same time, neural reconstruction is inherently limited to views close to the original driving log, as they do not have the ability to "imagine" previously unseen parts of the world (e.g., if the rolled out policy takes a left turn where the driving log turns right). Representing dynamic objects, e.g., vehicles, pedestrians, is particularly challenging: there is limited multi-view consistency signal for training the neural representation; and there are unobserved angles, e.g., the back of a vehicle that is behind ego in the log but overtakes ego in simulation.

Promising research directions include combining neural reconstruction with generative models that help to "fix" artifacts and imagine novel parts [82], as well as combining scene-level and object-level neural representations to handle dynamic actors [80, 83].

Generalizable neural reconstruction. Early neural rendering methods require per-scene optimization, i.e., they must be trained on images from a specific scenario and store model weights dedicated to that scenario. While this is still more efficient than manually creating 3D assets, generating scenes remains time- and storage-intensive. Recent advances in generalizable reconstruction models, such as pixelSplat [84], STORM [85], and others [86–89], predict a neural representation directly from input images using a feedforward network that generalizes over scenes, eliminating the need for per-scene optimization. However, reconstruction fidelity is usually slightly worse than for their per-scene optimized counterparts, especially for dynamic objects or extreme novel views. Improving generalizable reconstruction models is an important direction for future research.

2) How are traffic agents simulated?: Accurate simulation of traffic agents is essential for creating realistic driving scenarios. The fidelity of traffic models has critical safety implications: overly conservative agents that react too quickly or avoid collisions too effectively lead to optimistic performance evaluations, while non-reactive or overly aggressive agents produce pessimistic assessments. This challenge becomes particularly acute in CL training, where the ego policy can learn to exploit traffic model limitations, creating a substantial sim-to-real gap. For example, the policy may drive aggressively, knowing that surrounding reactive vehicles will always yield. Conversely, non-reactive agents can "leak" information about the expert behavior, e.g., an accelerating rear vehicle indicates an accelerating expert, leading to causal confusion in the learned policy.

Log-replay is the predominant traffic simulation approach, where recoreded agent trajectories are simply replayed during simulation. This approach has perfect fidelity in terms of agent-agent interactions; however, it ignores the actions of the ego vehicle, and thus it has limited applicability to ego states near the original recording.

Rule-based traffic models employ handcrafted agent behaviors that follow road geometry or recorded paths, such as IDM [90] or more sophisticated models [66, 91]. Rule-based methods generally struggle to scale to complex urban scenarios with dense agent interactions.

Finally, **learned traffic models**, trained on real-world driving data, represent the most flexible approach. These models can condition on HD maps, surrounding agent states, and ego vehicle states to generate realistic behaviors. Architectures include supervised learning approaches [92], conditional variational autoencoders [56, 93], motion forecasting Transformers [94–97], and diffusion models [98–101]. Recent advances leverage next-token prediction (NTP) models that tokenize actions [46, 47], and perform CL fine-tuning [45, 102] to enhance traffic model robustness, mirroring the ego policy training approaches examined in this survey.

Measuring the quality of traffic models also presents unique challenges for both training and evaluation, as traditional metrics like off-road and collision rates measure trajectory quality rather than behavioral realism or policy training utility. Two paradigms have emerged: distributional similarity to human drivers, which requires models to capture the stochastic nature of human behavior [103], and adversarial objectives that generate challenging scenarios to improve ego policy robustness [104]. The optimal balance between these objectives

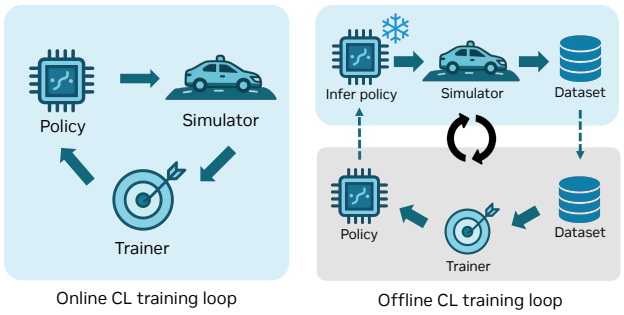


Fig. 8. **Online closed-loop training** with simulation inside the training loop (left) and **offline closed-loop training** that alternates between policy training and simulation rollout collection (right).

remains an open research question, as does determining how to effectively incorporate learned traffic models into CL training without overfitting to their limitations.

3) How are vehicle control and dynamics modeled?: When unrolling an ego policy during training, an important consideration is to what extent to simulate the downstream components present on the real vehicle (e.g., trajectory optimizer, controller), and the vehicle's dynamics model, including any latencies. Generally, the higher the dynamical fidelity the smaller the gap between training and deployment; but including downstream modules and accurate vehicle models in a training loop increases complexity and computation. Further, non-differentiable downstream components may be incompatible with some training frameworks, e.g., imitation learning, depending on the action space of the policy.

Simulation approaches exist on both ends of the spectrum, e.g., MagicDrive [105] and Vista [106] assume a "perfect" controller that **teleports** the AV to the new location according to the planned trajectory; AlpaSim [79] uses downstream controllers with a **simple vehicle model**; and DriveSim [67] simulates the full onboard system with **physics simulation** for vehicle dynamics.

4) How are scenarios generated?: A *scenario* in AV simulation refers to the arrangement of the scene in terms of road layout, state of traffic participants, ego state, visual appearance, etc. Scenarios are often obtained from **real-world logs**, optionally with small perturbations in case of neural reconstruction, or **constructed programmatically** in case of game engines. Alternatively, one may generate entire scenarios of interest using learned **generative models**, such as autoregressive models [107, 108], diffusion models [109, 110] and retrieval-augmented generation [111, 112]. Finally, there is a plethora of works on **adversarial scenario generation**, which optimizes scenarios to be more challenging for a given ego policy, applied to both AV testing and training [113]. Interesting research directions include measuring and improving realism, and improving controllability for long-tail scenarios, e.g., with natural language conditioning [111, 114, 115].

C. Generative video and world models

Recent advances in generative AI have made video generation [116] and interactive world model generation [117, 118] a promising alternative to geometrically grounded AV simulation. Early systems such as GAIA [119], MagicDrive [105], Cosmos-Drive [120], and others [26, 106, 121, 122] demonstrate the feasibility of this direction. Unlike game engines or neural reconstruction methods that follow a 3D-to-2D rendering pipeline, generative models synthesize 2D frames directly, typically using diffusion or autoregressive architectures. Their main advantage lies in flexibility: they can evolve scenes beyond the constraints of recorded logs and create rare, long-tail events of interest, potentially driven by natural language input.

However, the absence of explicit 3D grounding introduces challenges. Spatial and temporal consistency is fragile: vehicles may change appearance between frames or pass unrealistically through tight spaces. The lack of structured scene representations complicates evaluation and reward design, as collision boundaries, lane constraints, and other semantics are not explicitly defined. Additionally, AV simulation may require capabilities not natively supported by general-purpose video models: precise ego pose control, multi-view cameras, additional sensors. The modeling and controllability of traffic agents is also limited; while conditioning on object poses or road graphs has been explored, alignment remains imperfect. Notably, TeraSim [122] has recently shown promising results for generative video models to act as high-fidelity AV simulators by using StreetView images as conditioning and external traffic agent models.

Another promising direction is to combine geometrically grounded reconstruction and generative approaches. For example, SimGen [123] and others [124] use diffusion-based video generation to enhance realism in game engine renderings. Difix3D+ [82] combines neural rendering with video diffusion models to refine reconstructed images by correcting artifacts and filling in missing regions. Infinicube [125] integrates video generation with 3DGS to produce full 3D driving scenarios conditioned on HD maps. Such combinations leverage the controllability of structured simulation while benefiting from the realism and generative flexibility of video models.

D. Latent world models

Latent world models provide an alternative to sensor-space AV simulation by learning a forward dynamics model over policy features, i.e., predicting future latent states from past latent states and ego actions [57, 126]. They can be formulated as Bayesian probabilistic predictors [127, 128] or energy-based models such as JEPA [129, 130]. Notable examples in the context of AVs include [130–136]. They define latent states as intermediate policy features and train predictive models to minimize divergence between predicted and observed latent states, typically using KL divergence, L1, or L2 losses.

Like sensor-space simulation, latent models enable counterfactual rollouts beyond static datasets by evolving trajectories in latent space or perturbing latent states and actions. Their main advantage is avoiding the complexity and computational cost of high-fidelity sensor simulation, which often requires modeling details irrelevant for driving, such as foliage, shadows, reflections, or distant traffic agents. Instead, latent models

capture only features relevant to the policy, simplifying the problem and eliminating perceptual and behavioral sim-to-real gaps.

However, latent models are tightly coupled to the policy, requiring retraining and new data whenever the policy being trained changes. Their predictive quality is limited by the informativeness of the latent state, thus they are expected to struggle with long-tail scenarios much like the policy itself. In addition, reliable metrics for assessing prediction quality are still lacking, and inaccurate predictions can degrade performance or cause the policy to overfit to flawed latents akin to the sim-to-real gap in sensor simulation. Addressing these challenges remains an important research direction.

Despite the challenges, early results are promising: latent models have been used to optimize trajectories directly in latent space [130] and to constrain rollouts toward expert-like states [132]. We discuss these approaches further in Section VII-A2.

E. Open research directions

Beyond the method-specific research directions discussed in the preceding subsections, several overarching challenges remain for environment response generation in CL training.

A central concern is the persistent *sim-to-real gap*. Most rendering pipelines are optimized for human perception rather than for policy training, and the impact of simulation artifacts on policy training is poorly understood. Developing training techniques that mitigate the sim-to-real gap, as well as metrics for *policy-oriented fidelity*, are important open directions.

A consequence of the sim-to-real gap is that policies may overfit to the simulator, undermining that same simulator's ability to reliably evaluate the policy. This calls for new evaluation methodologies and benchmarks for *cross-simulator generalization*.

Orthogonal to simulation fidelity is *intelligent curriculum generation*: actively discovering policy weaknesses rather than relying on passive data collection. Promising research directions include integrated generative pipelines that combine LLMs for scenario specification, diffusion models for instantiation, and RL for adaptive execution, as well as constrained adversarial generation techniques that produce challenging yet plausible and solvable failure modes.

Finally, emerging paradigms such as generative video models and latent world models are rapidly evolving but remain relatively immature for CL training. Understanding the trade-offs and synergies between compact latent simulation and full sensor-space simulation is crucial for assessing and leveraging their long-term potential for CL policy training.

VII. How is the Training Objective Defined?

The training objective defines how CL experience is converted into a learning signal for updating the policy. The main paradigms are *closed-loop imitation learning* (CL-IL), the CL counterpart of the more common open-loop IL paradigm, which derives the signal from expert demonstrations in a CL setting, and *reinforcement learning* (RL), which derives the signal from a (handcrafted) reward function. *Hybrid* approaches combine elements of both, either by learning the reward function from demonstrations or by directly mixing IL and RL objectives. The conceptual spectrum is illustrated in Fig. 9.

Expert demonstrations from human drivers are widely regarded as a more robust way to specify desirable driving behavior than reward design. This is partly because AVs should *drive like humans*, but encoding human-likeness into a reward function remains notoriously difficult, particularly in rare or ambiguous situations [137, 138]. Consequently, (open-loop) IL remains the primary training mechanism in the AV industry today [9].

To mitigate the OL-CL gap, CL-IL techniques extend imitation objectives to CL policy rollouts. A key challenge is that pre-collected demonstrations only define desirable behavior along recorded trajectories, and not novel states encountered during CL rollouts. Since driving is inherently multi-modal, even an optimal policy may frequently and justifiably diverge from a single recorded trajectory. CL-IL methods address this by querying an expert on demand, or reusing fixed demonstrations with augmentations, guided rollouts, recovery heuristics, or latent-space objectives.

In contrast, RL uses a reward function that can be evaluated at any state, providing greater flexibility but introducing challenges in reward specification, optimization stability, and data efficiency. *Hybrid* methods combine the strengths of CL-IL and RL, either by learning a reward model from demonstrations or by integrating IL and RL objectives through sequential or parallel strategies.

The choice of training objective is also coupled with action and environment response generation. As discussed earlier, on-policy actions are the most effective in exposing the policy to the consequences of its own decisions, thereby reducing the OL–CL gap, but CL-IL methods that rely on pre-collected demonstrations may struggle with fully on-policy actions, as rollouts can quickly diverge from demonstrations. RL naturally supports on-policy rollouts but imposes stronger requirements on the environment: (i) high fidelity in all reachable states; (ii) ability to compute rewards, often requiring structured world states; and (iii) high throughput and low latency, since RL embeds simulation in the training loop.

In the following we review IL, RL, and hybrid approaches and discuss open research directions.

A. Closed-loop imitation learning

Closed-loop imitation learning (CL-IL) methods extend IL to the CL setting. The training objective for standard (open-loop) IL can be expressed as

$$\min_{\theta} \mathbb{E}_{(o_t^E, a_t^E)\sim\mathcal{D}} \, \ell\big(a_t^E, \pi_\theta(a_t \mid o_t^E)\big) \,,$$

using notation from Section II, where $\mathcal{D} = \{(o_t^E, a_t^E)\}$ is a fixed dataset of expert demonstrations collected under an expert policy $a_t^E \sim \pi_E(a_t \mid o_t^E)$, and ℓ denotes a supervised loss, such as mean squared error or cross-entropy.

The objective for CL-IL is then

$$\min_{\theta} \mathbb{E}_{(o_t^R, a_t^*)\sim\mathcal{D}_\theta^R} \, \ell\big(a_t^*, \pi_\theta(a_t \mid o_t^R)\big) \,.$$

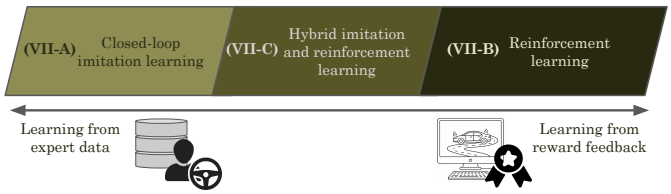


Fig. 9. **Training objectives** in terms of anchoring to expert demonstrations vs. handcrafted rewards.

The key difference is that training examples are sampled from a policy-induced rollout dataset $\mathcal{D}_\theta^R = \{(o_t^R, a_t^*)\}$, collected by rolling out the policy $a_t^R \sim \pi_\theta(a_t \mid o_t^R)$ following environment dynamics $o_{t+1}^R \sim f_d(o_{t+1} \mid a_t^R, o_t^R)$, and assigning action labels a_t^* to each encountered observation. The rollout dataset is typically re-collected throughout training. The policy used for rollouts is often modified to encourage exploration or guide towards pre-collected demonstrations. Sequence-level losses may also be included, enabling gradient backpropagation through time.

The central question in CL-IL is how to obtain the action label a_t^* for an observation o_t^R encountered during CL rollouts. Approaches can be broadly categorized into two groups: those that collect new demonstrations from an on-demand expert, $a_t^{R*} \sim \pi_E(a_t \mid o_t^R)$; and those that leverage pre-collected demonstrations by initializing rollouts from states in the dataset, $o_0^R \sim \mathcal{D}$, and deriving proxy expert actions from recorded trajectories, $a_t^* = f(o_t^R; o_{1:T}^E, a_{1:T}^E)$. We discuss these approaches in detail below.

1) Collecting new demonstrations: Classic methods like DAgger [28] and DART [139] assume the existence of an *on-demand expert* who can provide corrective demonstrations for any state once the policy deviates from the originally logged expert trajectory. Such an expert may be a human or an expert model.

On-demand human expert. Although large-scale human driving logs provide the main source of expert supervision for IL, it is difficult to collect human demonstrations for novel states induced by a policy rollout: first, due to the significant cost and time, as data collection has to be repeated as the policy changes; second, if using simulation, due to the difficulty of designing an effective human-computer interface; and third, due to the challenge of taking over control from the policy, requiring precise real-time intervention in a dynamic system like a vehicle. For these reasons, on-demand human expert demonstrations are rarely considered in the literature [140].

At the same time, disengagement events in deployed fleets present a potential opportunity: safety drivers and end-users already provide takeover demonstrations when the system fails. How to leverage these interventions remains underexplored [60, 141, 142].

Expert model. New expert demonstrations can also be provided via an "expert" model or algorithm, which are more scalable, and potentially also more consistent and risk-averse, than human drivers.

A common choice is a *privileged policy* with access to structured world states such as HD maps and object bounding boxes [40, 143–145]. Leveraging RL to train a privileged policy and use it as an expert teacher is particularly promising (see Section VII-C2). Downsides include the requirement for environment-provided privileged information and the complexity of developing the expert model itself.

Alternative end-to-end systems, such as modular AV stacks, can also serve as experts. While using one autonomous system to train another may seem circular, modular AV stacks can exhibit strong robustness in specific operational domains (e.g., lane keeping) and can therefore be leveraged in a targeted manner to strengthen the policy being trained.

Finally, expert demonstrations can also be generated in latent space, using a world model that implicitly encodes desirable driving. We discuss latent model-based imitation in Section VII-A2.

2) Leveraging pre-collected demonstrations: To overcome the cost and scalability bottlenecks of querying an online expert, a second family of approaches generates additional data and supervision by cleverly re-using pre-collected expert demonstrations.

Handcrafted recovery. Heuristics and handcrafted controllers can generate "recovery" trajectories that return the ego vehicle to the pre-collected expert trajectory, e.g., geometric controllers for lane keeping [146] or PID controllers for following distance [147]. With discrete action spaces, recovery can simply select the action that realigns with the demonstration [45]; in continuous domains, trajectory optimization can target an expert state as the terminal condition [148]. These strategies can be effective for small deviations, but as the state drifts further from demonstrations, recovery trajectories may cease to represent desirable driving behavior and can even degrade policy performance, e.g., if recovery induces a collision or if the rollout and expert trajectory correspond to different valid modes of the driving distribution.

Guided rollouts. Guided rollouts extend the applicability of recovery heuristics by biasing action selection toward states that remain closer to a fixed expert trajectory. A notable example is the Closest Among Top-K (CAT-K) strategy [45], and its extension to end-to-end driving in RoaD [149], which considers K action samples from the policy at each rollout step, and executes the one that yields a future state closest to the expert trajectory. This balances on-policy action generation with the continued validity of expert demonstrations as a training target. Open challenges include backpropagating over multiple time step, and distance metrics for CAT-K selection beyond Euclidean distance, which is ill-suited for low-velocity and ambiguous maneuvers.

History perturbation. Another simple approach takes states along pre-collected demonstrations and perturbs only the state history fed into the policy [150]. By creating only small deviations in the input, the original expert action can be used as valid training targets, teaching the policy to recover from minor errors [44, 151, 152]. A key limitation is that the CL experience is generated heuristically rather than through policy rollouts, and exposing the policy to all relevant failure modes requires careful perturbation design and extensive tuning.

Latent model-based IL. Latent world models (Section VI-D) treat policy features as latent states and predict future latents from past latents and ego actions. Jointly training

the policy and world model, e.g., MILE [131] or LAW [134], provides a richer training signal: beyond actions the policy also learns to predict latents derived from future observations.

Crucially, Popov et al. [132] show that latent world models enable a form of CL training that directly mitigates covariate shift: the world model acts as a differentiable simulator, allowing gradients to backpropagate through time from a similarity loss defined between predicted and expert-derived latents.

While promising for reducing the OL-CL gap, open questions remain. Expert-derived latents lose validity once rollouts drift far from demonstrations, similar to observation-space CL training, and latent representations may overlook critical scene aspects, such as interactive behavior of other traffic participants. Synergies and trade-offs between latent world models and state-space CL imitation merit further investigation.

B. Reinforcement learning

Reinforcement learning (RL) derives its training signal from a reward function that can be evaluated for any state encountered during CL rollouts. This allows the agent to explore novel states, receive direct feedback from the environment, and gradually adapt its behavior to maximize cumulative reward, without requiring new expert annotations. In doing so, RL offers a pathway to move beyond expert demonstrations and develop robust strategies for rare or highly interactive scenarios.

RL algorithms can be broadly categorized into *policy-based* methods, which directly parameterize and optimize the policy; *value-based* methods, which learn action-value functions $Q(o, a)$ and derive actions by maximizing estimated values; and *actor-critic* methods, which combine both by learning a policy (actor) alongside a value function (critic). Most RL work in autonomous driving focuses on policy gradient and actor-critic approaches, which we formalize below.

Policy-based methods optimize the expected *return* (i.e., total discounted reward) along policy rollouts,

$$J(\theta) = \mathbb{E}_{r\sim p_\theta(r)}\left[\sum_{t=1}^{T} \gamma^{t-1} R'(o_t, a_t)\right],$$

where γ is the discount factor and CL rollouts $r = (o_1, a_1, \ldots, o_T, a_T)$ are generated from the policy $a_t \sim \pi_\theta(a_t \mid o_t)$ and the environment dynamics $o_{t+1} \sim f_d(o_{t+1} \mid o_t, a_t)$. Since a "true" reward function R is not generally known, R', a proxy reward function is chosen. The policy parameters θ are updated in the direction of the *policy gradient*,

$$\nabla_\theta J(\theta) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a_t \mid o_t)\, A_t],$$

where A_t denotes an *advantage* term estimating how much better the taken action performed than expected. Actor-critic methods augment this formulation by learning a value function $V^\pi(o_t)$ that predicts the expected return from observation o_t, which is used to compute the advantage as $A_t = \sum_{k=t}^{T} \gamma^{k-t} R'(o_k, a_k) - V^\pi(o_t)$, where the first term is the discounted sum of future rewards.

Applying RL to end-to-end driving, however, presents unique challenges. The reward function must encode nuanced, human-like behavior even in edge cases. This specification problem is notoriously difficult, often motivating hybrid approaches that bring back dependency on expert demonstrations. Further challenges include requiring a high-quality and fast simulation environment, compatible policy architecture and action representation, and learning accurate value estimates.

In the following we review algorithmic choices, distinguishing actor-critic methods (Section VII-B1) and critic-free methods (Section VII-B2), and discuss applying RL to the task of autonomous driving (Section VII-B3). We defer hybrid approaches that learn rewards from demonstrations or combine IL and RL objectives to Section VII-C.

1) Actor-critic algorithms: Most modern RL algorithms follow the *actor-critic* paradigm, combining an *actor* (policy) that selects actions with a *critic* (value function) that provides variance-reducing baselines for policy gradient updates. The critic computes an *advantage* signal quantifying how much better each action is compared to what the critic predicts the expected return would be from that state. Estimating the advantage accurately is crucial for stable learning, necessiating techniques like Generalized Advantage Estimation [153] to balance bias and variance.

Algorithms further differ in terms of the *on-policiness* of their action generation approach (Fig. 5), which determines how efficiently they reuse data and how stably they update the policy. **On-policy** algorithms such as PPO [53] and TRPO [154] collect rollouts from the current policy, π_θ, and use those fresh trajectories to compute stable updates. **Off-policy** algorithms such as SAC [50], DDPG [155], or TD3 [156] can (re-)use experience generated by a different variant of the policy, $\pi'_\theta \sim \pi_\theta$, typically earlier versions of the same policy (a "stale" policy), though other rollout strategies as described in Section V can also be employed. Reusing experience dramatically improves sample efficiency; however, correcting for the mismatch between previously collected data and the current policy introduces bias/variance challenges that can destabilize training, necessitating techniques such as clipped importance ratios [52], conservative updates [154], double critics [156], prioritized experience replay [157], and others.

At the extreme end of off-policy learning, **offline RL** trains from a fixed dataset without further environment interaction. Offline RL is closely related to open-loop IL in that both rely on a fixed dataset, but offline RL optimizes for cumulative reward rather than imitating an expert, allowing consideration of longer-term consequences of actions in CL. Nevertheless, without exposure to actual CL rollouts, offline RL remains vulnerable to distributional shift and the broader OL–CL challenge discussed in Section III-B.

2) Critic-free algorithms: In autonomous driving, learning a reliable value function can be challenging (more in Section VII-B3). Critic-free algorithms, such as REINFORCE [158], directly optimize the expected return *without* learning a value function, and have been successfully applied to traffic simulation [159]. However, because they lack a critic to provide variance-reducing baselines, these methods usually suffer from high-variance policy gradients and are mostly

restricted to on-policy training, which limits sample efficiency and slows convergence.

Advances in Large Language Models (LLMs) have led to sophisticated critic-free methods such as DPO [160], GRPO [161] and others [162–164], which replace value estimation with normalization across rollouts. These approaches show promise for driving, with early successes in recent work [165–167].

3) Applying RL to autonomous driving: There is a long history of research on using RL for autonomous driving. Most prior work, however, focuses on *vectorized* simulation (comprised of bounding boxes, polyline HD maps, etc.), which sidesteps the issue of perception and visual rendering [168–172]. Others use game-engine simulators like CARLA [66], which offer visual observations but limited behavioral complexity due to predictable traffic agents and lack of realistic comfort or human-likeness metrics [145, 173–175]. Recent advances in reconstruction-based simulation show promise for more realistic RL-based driving [51]. Note that many successful works combine IL and RL through sequential training or expert-based rewards (see Section VII-C).

Despite significant advances in RL research, widespread adoption for industry-scale, deployable AVs remains limited by several key challenges and design considerations, which we detail below.

a) Designing a reward function. The reward function dictates policy behavior, but human-like driving is highly nuanced: it requires balancing safety, efficiency, comfort, and adherence to traffic regulations, while capturing subtle behaviors in rare interactive scenarios. Many prior works rely on a weighted sums of handcrafted terms [176–178]. However, such formulations may miss rare but important scenarios; linear weights cannot represent nuanced trade-offs; and handcrafted comfort terms may not capture human preferences in real-world driving. Alternative approaches learn rewards from demonstrations or preferences (Section VII-C1) or penalize deviation from expert trajectories (Section VII-C2).

Reward specification is related to evaluation metric design, with a key distinction that RL also depends on *reward shaping*, i.e., the density, informativeness, and predictability of rewards. It is well known that even minor misspecifications (in edge cases) can cause *reward hacking*, unsafe shortcuts, or brittle policies [179]. Much of the literature sidesteps the issue of reward realism by using the same custom reward for training and evaluation, or tailoring reward terms to benchmark metrics [31, 180]. How to evaluate reward functions themselves, both in terms of fidelity to desired behavior and thier suitability as an optimization objective, remains a key open question.

b) RL-compatible simulation environment. RL imposes stricter demands on the environment than CL imitation. First, it requires high-fidelity simulation even in unseen states that are far from logged trajectories, in order to prevent overfitting to simulation artifacts (Section VI). Fidelity is crucial not only for visual observations but also for the traffic agent behavior, as RL is prone to exploiting unrealistic or predictable interactions (Section VI-B2). Second, RL often relies on structured world state (e.g., HD maps, bounding boxes) to compute handcrafted rewards.

Finally, RL also requires high-throughput simulation due to its lower data efficiency. For on-policy methods, latency (including reconfiguration time between scenes) becomes critical since the simulator is embedded in the training loop (Fig. 8). To provide a sense of scale, prior work in simplified settings used PPO with 300M steps in CARLA, 500M in nuPlan [172], and 2.5B steps in JAX-accelerated simulation [170]. This translates to 10^4–10^5 steps per second to complete training in 1 week, which remains a major challenge for end-to-end simulation.

c) RL-compatible policy architecture and action space. Unlike CL-IL, RL introduces considerations for policy architecture and action space. While RL in other domains often employs small and simple networks, driving policies are necessarily larger and more complex, as they must jointly address 3D perception and interaction modeling. Consequently, conclusions from standard RL benchmarks may not directly translate to autonomous driving.

The choice of action space also requires attention. RL generally assumes instantaneous actions along with log-probabilities to compute policy gradients. Instead, a common choice in driving is to predict an extended-horizon *trajectory* (Section II). Treating trajectories as instantaneous actions inflates dimensionality and variance as most action dimensions do not affect the return. For tokenized policies, that discretize trajectories into tokens [13, 46, 181], the tokens can be treated as discrete actions; however, longer trajectories yield multiple tokens with equivalent instantaneous effects, and a trajectory may be encoded into multiple tokens, complicating the mapping to actions. Finally diffusion or flow matching policies, such as [13, 182], do not directly provide log-probabilities for the generated actions, requiring specialized RL algorithms [183]. Most prior RL studies sidestep these difficulties by adopting simplified action spaces for RL experiments, though these may not scale to real-world systems. Effective handling of alternative action representations therefore remains an important open direction.

d) Value function estimation. Actor–critic methods require learning a value function, which poses challenges in driving. First, the value function needs a large network to address perception. Asymmetric actor–critic methods that use privileged value functions (with structured world state input) offer one remedy [184]. Second, pre-trained policies usually lack natural value estimators, and warm-starting RL with a poorly trained critic can lead to instability [185] (see Section VII-C2). Finally, predicting long-term returns is difficult when optimal returns depend on stochastic dynamics induced by other traffic participants, so reward design must also consider the predictability of returns.

C. Hybrid imitation and reinforcement learning

The most promising CL training approaches combine elements of imitation and RL, trading some dependence on expert demonstrations for improved training efficiency. We categorize hybrid approaches into two main groups: methods that derive rewards from expert demonstrations (Section VII-C1), and methods that directly combine IL and RL objectives (Section VII-C2).

1) Imitation-based rewards for RL: The first group of methods grounds the RL objective in expert demonstrations, while leveraging RL algorithms to optimize the policy toward this objective. Approaches in this category include adding imitation-based reward terms, applying inverse RL to learn rewards from demonstrations, or adversarial imitation that trains a discriminator to distinguish expert from policy behavior and then update the policy via policy gradients.

a) Deviation from expert reward term. A straightforward approach augments the handcrafted reward with a term penalizing distance between rollout states and expert demonstrations, effectively transforming the IL objective into a reward function. While RL is less sample-efficient than the direct supervised regression of IL, it enables reasoning about delayed consequences and combining imitation with other reward terms (safety, progress, etc.). However, imitation rewards may yield returns that are difficult to predict, resulting in inaccurate value estimates and consequently unstable or slow training.

b) Learned reward function / inverse RL. Inverse RL methods first learn a reward function from expert demonstrations, such that under the learned rewards the expert demonstrations are (soft-)optimal [186]. The learned reward function is then used to derive a policy via planning or via subsequent RL [187].

In autonomous driving works differ in whether they learn a reward function from structured world states (lanes, objects, etc.) [188–190] or from raw sensor inputs [191, 192]. By grounding rewards in demonstrations, this approach ties RL to the only direct evidence of desirable driving behavior, similar to IL, while still enabling temporal CL optimization with RL. However, compared to directly learning a generalizable policy as in IL, the formulation shifts the complexity to learning a generalizable reward function, which is a more indirect representation. Evaluating the accuracy and utility of learned rewards remains an open challenge [193, 194].

Advances in LLMs have renewed interest in *learning from preferences*, where experts provide relative rankings rather than demonstrations. Preferences can be used to learn a reward function for subsequent RL [195, 196], or to directly optimize a policy [160]. In the context of autonomous driving, recent examples include [197–199]. One advantage of preferential learning is that ranking simulated CL rollouts is often more feasible for human experts than taking over vehicle control to provide corrective actions. However, scalability and the design of suitable human–computer interfaces for reliably evaluating nuanced driving outcomes remain key challenges [200, 201].

c) Adversarial imitation learning. Adversarial imitation learning methods, such as GAIL [202] and related works [203–206], formulate IL as distribution matching between CL trajectories generated by the expert and the policy. A discriminator is trained simultaneously to distinguish expert from policy state–action pairs, and its output serves as a surrogate reward for updating the policy via RL. This formulation sidesteps the explicit recovery of a reward function, and directly drives the policy to match the expert's CL trajectory distribution. Adversarial imitation has been applied to end-to-end driving, although mostly only using game-engine based simulation [66, 203, 207–209].

Adversarial IL reduces covariate shift compared to BC, and its discriminator provides a denser reward signal for RL than most handcrafted reward functions. However, it may suffer from the same sample inefficiency and dependency on high-quality simulation as standard RL; as well as instability and mode collapse in the discriminator, similar to GANs. Improving discriminators for high-dimensional vision and integrating safety constraints are important future directions.

2) Combined IL and RL objectives: The second group integrates IL and RL objectives through sequential training (IL pre-training followed by RL fine-tuning), parallel optimization of both losses, using an IL policy to guide RL, or teacher–student schemes where a privileged RL teacher is distilled into an end-to-end student via IL.

a) IL pre-training and RL finetuning. A promising paradigm pre-trains the policy with BC then fine-tunes it in CL, similar to the pre-training and alignment stages in LLMs. However, warm-starting RL from IL is known to introduce several pitfalls that require careful IL-to-RL handoff strategies.

One challenge is poorly calibrated value estimators at the onset of RL training. If the critic is trained on static data, it often overfits to the behavior policy. When evaluated on CL rollouts it must extrapolate, causing *under- or overestimation bias* which then leads to policy divergence [210]. This issue is closely related to the "deadly triad" of function approximation, bootstrapping, and off-policy learning [211]. Remedies include conservative critics [212], advantage filtering [213, 214], and double Q-learning [215]. Alternatively, critic-free RL sidesteps value estimation entirely (Section VII-B2). Indeed many successful sequential IL+RL approaches in driving relying on critic-free methods [159], while advanced critic-regularization strategies remains underexplored in driving.

A second challenge is that imitation pre-training often produces nearly deterministic policies, causing RL to become trapped in local optima [50]. Conversely, large policy updates risk catastrophi, particularly when the OL-to-CL gap is large or when rewards conflict with expert demonstrations. While investigated in the broader RL literature [185, 211], solutions remain largely unexplored in driving [9, 23].

b) Parallel IL and RL. Jointly optimizing imitation and RL objectives encourages human-like behavior while improving CL robustness. Algorithmic variants include mixing expert demonstrations into replay buffers [216], adding imitation losses to policy-gradient methods [217], demonstration-based regularization [218], and advantage-weighted cloning that biases updates toward expert-like actions [219].

In autonomous driving, most works use simple combinations: Lu et al. [220] mix BC and RL examples in the same batch; and Gao et al. [51] alternate between RL and IL updates. More sophisticated algorithmic designs remain an open direction.

c) Imitation guided RL. Some approaches sidestep IL-to-RL transfer difficulties by using the IL policy as a guide rather than initialization for RL. For instance, RL can be trained from scratch while regularized toward an IL prior through shared perception modules [221], KL divergence constraints [222], or hierarchical formulations with IL-trained low-level skills [223,

224]. These strategies preserve IL benefits while avoiding instability from directly fine-tuning imitation-initialized policies.

d) Teacher–student scheme. Hybrid IL-RL also appears in teacher-student training schemes. A promising setup is where the teacher policy has access to privileged information, such as HD maps, object states, etc., and can therefore be trained more efficiently and robustly with RL. The privileged teacher then serves as an expert to generate demonstrations for a student policy that only observes raw sensor inputs [225]. Critically, unlike human experts, the teacher policy can be queried on demand, enabling DAgger-style CL imitation that mitigates covariate shift.

A notable early example of this framework in driving is Roach [145]. While the original experiments were limited to CARLA with relatively simplistic and predictable traffic agents, the approach could be extended to combine demonstrations from both privileged teachers and human drivers, aiming to learn human-like driving that also remains robust in CL execution. For training stronger privileged experts, multi-agent self-play provides a particularly promising direction, with recent results showing strong performance in privileged driving settings [226].

D. Open research questions

Despite rapid progress, CL training objectives for end-to-end driving remain underdefined. Objectives must simultaneously capture human preferences and safety, remain effective as rollouts deviate from demonstrations and encouter edge cases, enable stable and efficient optimization, and account for imperfect simulation. The field has yet to converge on a broadly applicable framework.

For CL imitation, balancing expert anchoring with exploration remains a key question. Strong imitation losses preserve human-like behavior but can hinder discovery; on-policy training promotes robustness to distribution shift but risks unsafe or unnatural actions when adhering to fixed demonstrations. Principled objectives that adapt expert guidance strength without introducing bias remain an important open problem.

For RL, reward design remains difficult: objectives must reflect the nuanced, multi-objective nature of driving, generalize across diverse environments, and resist reward hacking. Inverse RL and preference-based methods promise improved alignment, but scalability of reward learning and supervision cost of policy-specific human feedback persist. Progress likely requires holistic frameworks co-designing the learning algorithm with policy architecture, action space, and simulator limitations.

Another frontier in RL is scaling infrastructure from simplified academic settings to industry-scale, deployable end-to-end driving policies, which will require, among others, distributed frameworks with tight orchestration between trainers and simulators [227, 228].

Towards full autonomy, scalable policy training will likely adopt multi-stage pipelines: large-scale OL IL pre-training followed by CL fine-tuning integrating CL-IL and RL. A key challenge is stabilizing the OL-to-CL transition to prevent catastrophic forgetting, i.e., improving in specific scenarios without regressing on others. Another key challenge is advancing beyond naive loss mixtures toward principled objectives where IL and RL interact synergistically, e.g., such that RL dominates when imitation loses validity, and conversely, IL grounds RL in human priors to mitigate reward hacking.

VIII. Conclusions

This survey framed the training of end-to-end driving policies as the challenge of *closing the OL–CL gap*. Despite its fundamental importance for robust AV deployment, we argue that this perspective has not received sufficient attention in the literature or in industry practice. We organized CL training approaches into a taxonomy along three interdependent axes (action generation, environment response, and training objective) and analyzed the trade-offs within and across these dimensions, offering a unifying lens to interpret existing methods.

Recent advances have made CL training of end-to-end policies practically feasible, yet bringing together simulation, learning algorithms, and policy architectures into cohesive training pipelines remains an open challenge. We highlight several key directions toward this goal. First, *RL algorithms* must evolve to support modern end-to-end simulation, large vision-language-action policies with tokenized or diffusion-based outputs, and rich, human-like reward functions. Concurrently, *CL imitation learning* remains relatively underexplored in the context of end-to-end driving and offers a promising human-grounded and compute-efficient path. Second, *reconstruction-based AV simulation* must become training-oriented: current benchmarks focus predominantly on evaluation, and developing simulation optimized for policy learning, with appropriate fidelity, throughput, and latency, is crucial. Third, *sim-to-real transfer* deserves more attention: driving requires not only hard-to-achieve visual realism but also behaviorally faithful traffic agents to prevent policies from exploiting unrealistic interactions. Fourth, *video generative models* and *latent world models* represent emergent technologies with significant potential as training environments, but their utility for policy learning is only beginning to be explored. Finally, *principled hybrid approaches* could help reconcile the conflicting design choices identified in this survey into unified CL training frameworks.

Across these directions, a unifying theme emerges: the necessary technologies (neural rendering, generative world models, scalable RL) have matured largely in isolation, and we are now at a moment where they can be unified. Similar to how pre-training and alignment transformed large language models, we anticipate that OL pre-training followed by CL fine-tuning will become the standard for end-to-end driving. We hope this survey helps bridge the simulation, learning, and autonomous driving communities toward this goal.

Acknowledgments

The authors used AI tools (ChatGPT, Gemini, Claude) for language editing and assistance with certain figure elements. All technical content and conclusions were developed, verified, and approved by the authors.

REFERENCES

[1] Y. Zheng, Z. Xia, Q. Zhang, T. Zhang, B. Lu, X. Huo *et al.*, "Preliminary investigation into data scaling laws for imitation learning-based end-to-end autonomous driving," *arXiv preprint arXiv:2412.02689*, 2024. 1

[2] M. Baniodeh, K. Goel, S. Ettinger, C. Fuertes, A. Seff, T. Shen *et al.*, "Scaling laws of motion forecasting and planning–a technical report," *arXiv preprint arXiv:2506.08228*, 2025. 4

[3] A. Naumann, X. Gu, T. Dimlioglu, M. Bojarski, A. Degirmenci, A. Popov *et al.*, "Data scaling laws for end-to-end autonomous driving," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025, pp. 2571–2582. 1

[4] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani *et al.*, "Deep reinforcement learning for autonomous driving: A survey," *Trans. Intell. Transp. Syst. (T-ITS)*, vol. 23, no. 6, pp. 4909–4926, 2021. 1, 2

[5] General Motors, "Self-driving safety report," 2018, Available at https://www.gm.com/content/dam/company/docs/us/en/gmcom/gmsafetyreport.pdf. 1

[6] Waymo, "Safety report," 2021, Available at https://waymo.com/safety/safety-report. Retrieved on July 4, 2021.

[7] Motional, "Voluntary safety self-assessment," 2021, Available at https://drive.google.com/file/d/1JjfQByU_hWvSfkWzQ8PK2ZOZfVCqQGDB/view.

[8] P. Karkus, B. Ivanovic, S. Mannor, and M. Pavone, "Diffstack: A differentiable and modular control stack for autonomous vehicles," in *Conf. Robot Learn. (CoRL)*, 2022. 1, 6

[9] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 2024. 1, 2, 10, 14

[10] Wayve, "Pioneering the end-to-end ai driving model," 2025, Available at https://wayve.ai/technology/#AV2.0.

[11] NVIDIA, "Autonomous vehicles safety report," 2025, Available at https://images.nvidia.com/aem-dam/en-zz/Solutions/auto-self-driving-safety-report.pdf.

[12] A. Elluswamy, "Tesla's approach to autonomy," X (formerly Twitter), 2025, accessed: 2025-12-02. [Online]. Available: https://x.com/aelluswamy/status/1981644831790379245

[13] NVIDIA, Y. Wang, W. Luo, J. Bai, Y. Cao, T. Che *et al.*, "Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail," *arXiv preprint arXiv:2511.00088*, 2025. 1, 13

[14] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE access*, vol. 8, pp. 58 443–58 469, 2020. 2

[15] L. Chen, Y. Li, C. Huang, B. Li, Y. Xing, D. Tian *et al.*, "Milestones in autonomous driving and intelligent vehicles: Survey of surveys," *Trans. Intell. Veh. (T-IV)*, vol. 8, no. 2, pp. 1046–1056, 2022. 2

[16] R. Fan, S. Guo, and M. J. Bocus, "Autonomous driving perception," *Springer*, 2023. 2

[17] L. Wang, M.-A. Lavoie, S. Papais, B. Nisar, Y. Chen, W. Ding *et al.*, "Deployable and generalizable motion prediction: Taxonomy, open challenges and future directions," *arXiv preprint arXiv:2505.09074*, 2025. 2

[18] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *Trans. Intell. Veh. (T-IV)*, vol. 1, no. 1, pp. 33–55, 2016. 2

[19] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *J. Field Robot. (JFR)*, vol. 37, no. 3, pp. 362–386, 2020. 2

[20] P. S. Chib and P. Singh, "Recent advancements in end-to-end autonomous driving using deep learning: A survey," *Trans. Intell. Veh. (T-IV)*, vol. 9, no. 1, pp. 103–118, 2023. 2

[21] Y. Guan, H. Liao, Z. Li, J. Hu, R. Yuan, G. Zhang *et al.*, "World models for autonomous driving: An initial survey," *Trans. Intell. Veh. (T-IV)*, 2024. 2

[22] H. Gao, Z. Wang, Y. Li, K. Long, M. Yang, and Y. Shen, "A survey for foundation models in autonomous driving," *arXiv preprint arXiv:2402.01105*, 2024. 2

[23] Z. Zhu and H. Zhao, "A survey of deep rl and il for autonomous driving policy learning," *Trans. Intell. Transp. Syst. (T-ITS)*, vol. 23, no. 9, pp. 14 043–14 065, 2021. 2, 14

[24] L. Le Mero, D. Yi, M. Dianati, and A. Mouzakitis, "A survey on imitation learning techniques for end-to-end autonomous vehicles," *Trans. Intell. Transp. Syst. (T-ITS)*, vol. 23, no. 9, pp. 14 128–14 147, 2022. 2, 4

[25] J. Yang, B. Ivanovic, O. Litany, X. Weng, S. W. Kim, B. Li *et al.*, "Emernerf: Emergent spatial-temporal scene decomposition via self-supervision," *arXiv preprint arXiv:2311.02077*, 2023. 2

[26] L. Russell, A. Hu, L. Bertoni, G. Fedoseev, J. Shotton, E. Arani *et al.*, "Gaia-2: A controllable multi-view generative world model for autonomous driving," *arXiv preprint arXiv:2503.20523*, 2025. 2, 9

[27] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse control tasks through world models," *Nature*, pp. 1–7, 2025. 2

[28] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Int. Conf. Artif. Intell. Stat. (AISTATS)*, 2011, pp. 627–635. 3, 11

[29] J. Spencer, S. Choudhury, A. Venkatraman, B. Ziebart, and J. A. Bagnell, "Feedback in imitation learning: The three regimes of covariate shift," *arXiv preprint arXiv:2102.02872*, 2021. 3

[30] J.-T. Zhai, Z. Feng, J. Du, Y. Mao, J.-J. Liu, Z. Tan *et al.*, "Rethinking the open-loop evaluation of end-to-end autonomous driving in nuscenes," *arXiv preprint arXiv:2305.10430*, 2023. 3

[31] D. Dauner, M. Hallgarten, A. Geiger, and K. Chitta, "Parting with misconceptions about learning-based vehicle motion planning," in *Conf. Robot Learn. (CoRL)*, 2023, pp. 1268–1281. 13

[32] C. Zhang, R. Guo, W. Zeng, Y. Xiong, B. Dai, R. Hu *et al.*, "Rethinking closed-loop training for autonomous driving," in *Eur. Conf. Comput. Vis. (ECCV)*, 2022, pp. 264–282. 3, 6

[33] P. De Haan, D. Jayaraman, and S. Levine, "Causal confusion in imitation learning," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019. 3

[34] C. Wen, J. Lin, T. Darrell, D. Jayaraman, and Y. Gao, "Fighting copycat agents in behavioral cloning from observation histories," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, pp. 2564–2575, 2020. 3

[35] Y. Chen, P. Karkus, B. Ivanovic, X. Weng, and M. Pavone, "Tree-structured policy planning with learned behavior models," in *Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 7902–7908. 3, 6

[36] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?" *Transportation research part A: policy and practice*, vol. 94, pp. 182–193, 2016. 3

[37] H. X. Liu and S. Feng, "Curse of rarity for autonomous vehicles," *Nature Commun.*, vol. 15, no. 1, p. 4808, 2024. 3

[38] M. Bojarski, C. Chen, J. Daw, A. Değirmenci, J. Deri, B. Firner *et al.*, "The nvidia pilotnet experiments," *arXiv preprint arXiv:2010.08776*, 2020. 4, 7

[39] W. Yao, Z. Li, S. Lan, Z. Wang, X. Sun, J. M. Alvarez *et al.*, "Drivesuprim: Towards precise trajectory selection for end-to-end planning," *arXiv preprint arXiv:2506.06659*, 2025. 4

[40] A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger, "Exploring data aggregation in policy learning for vision-based urban autonomous driving," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 11 763–11 773. 6, 11

[41] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," in *Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2023.

[42] B. Jaeger, K. Chitta, and A. Geiger, "Hidden biases of end-to-end driving models," in *Int. Conf. Comput. Vis. (ICCV)*, 2023.

[43] Z. Li, W. Yao, Z. Wang, X. Sun, J. Chen, N. Chang *et al.*, "Generalized trajectory scoring for end-to-end multimodal planning," *arXiv preprint arXiv:2506.06664*, 2025. 4

[44] M. Bansal, A. Krizhevsky, and A. Ogale, "ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst," in *Robot.: Sci. Syst. (RSS)*, 2019. 4, 6, 11

[45] Z. Zhang, P. Karkus, M. Igl, W. Ding, Y. Chen, B. Ivanovic *et al.*, "Closed-loop supervised fine-tuning of tokenized traffic models," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025. 6, 8, 11

[46] J. Philion, X. B. Peng, and S. Fidler, "Trajeglish: Traffic modeling as next-token prediction," in *Int. Conf. Learn. Represent. (ICLR)*, 2024. 8, 13

[47] W. Wu, X. Feng, Z. Gao, and Y. Kan, "Smart: Scalable multi-agent real-time simulation via next-token prediction," *Neural Inf. Process. Syst. (NeurIPS)*, 2025. 8

[48] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 9329–9338. 6

[49] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013. 6

[50] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 1861–1870. 12,

14

[51] H. Gao, S. Chen, B. Jiang, B. Liao, Y. Shi, X. Guo *et al.*, "Rad: Training an end-to-end driving policy via large-scale 3dgs-based reinforcement learning," *arXiv preprint arXiv:2502.13144*, 2025. 6, 13, 14

[52] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward *et al.*, "IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures," in *Int. Conf. Mach. Learn. (ICML)*, 2018. 6, 12

[53] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017. 6, 12

[54] M. Igl, D. Kim, A. Kuefler, P. Mougin, P. Shah, K. Shiarlis *et al.*, "Symphony: Learning realistic and diverse agents for autonomous driving simulation," in *Int. Conf. Robot. Autom. (ICRA)*, 2022, pp. 2445–2451. 6

[55] Z. Huang, P. Karkus, B. Ivanovic, Y. Chen, M. Pavone, and C. Lv, "Dtpp: Differentiable joint conditional prediction and cost evaluation for tree policy planning in autonomous driving," in *Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 6806–6812. 6

[56] D. Xu, Y. Chen, B. Ivanovic, and M. Pavone, "Bits: Bi-level imitation for traffic simulation," in *Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 2929–2936. 6, 8

[57] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt *et al.*, "Mastering Atari, Go, Chess and Shogi by planning with a learned model," *arXiv preprint arXiv:1911.08265*, 2019. 6, 9

[58] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 2052–2062. 6

[59] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa *et al.*, "Continuous control with deep reinforcement learning," in *Int. Conf. Learn. Represent. (ICLR)*, 2016. 6

[60] W. Zhou, B. Zhang, Z. Cao, X. Li, Q. Cheng, C. Liu *et al.*, "Drarl: Disengagement-reason-augmented reinforcement learning for efficient improvement of autonomous driving policy," *arXiv preprint arXiv:2506.16720*, 2025. 7, 11

[61] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. M. Wolff, A. H. Lang *et al.*, "NuPlan: A closed-loop ml-based planning benchmark for autonomous vehicles," in *Conf. Comput. Vis. Pattern Recognit. Worksh. (CVPRW)*, 2021. 7

[62] C. Gulino, J. Fu, W. Luo, G. Tucker, E. Bronstein, Y. Lu *et al.*, "Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, pp. 7730–7742, 2023. 7

[63] S. Kazemkhani, A. Pandya, D. Cornelisse, B. Shacklett, and E. Vinitsky, "Gpudrive: Data-driven, multi-agent driving simulation at 1 million fps," *arXiv preprint arXiv:2408.01584*, 2025. 7

[64] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 1, 1988. 7

[65] L. Feng, Y. Gao, E. Zablocki, Q. Li, W. Li, S. Liu *et al.*, "Rap: 3d rasterization augmented end-to-end planning," *arXiv preprint arXiv:2510.04333*, 2025. 7

[66] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Conf. Robot Learn. (CoRL)*, 2017. 7, 8, 13, 14

[67] NVIDIA, "Drivesim," 2025, accessed: 2025 August 11. [Online]. Available: https://developer.nvidia.com/drive/simulation 8, 9

[68] Q. Li, Z. Peng, L. Feng, Q. Zhang, Z. Xue, and B. Zhou, "Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning," *Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 2022. 8

[69] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2232–2241. 8

[70] S. Wang, D. Jia, and X. Weng, "Deep reinforcement learning for autonomous driving," *arXiv preprint arXiv:1811.11329*, 2018. 8

[71] J. Gao, T. Shen, Z. Wang, W. Chen, K. Yin, D. Li *et al.*, "Get3d: A generative model of high quality 3d textured shapes learned from images," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 35, pp. 31 841–31 854, 2022. 8

[72] P. Cai, Y. Lee, Y. Luo, and D. Hsu, "Summit: A simulator for urban driving in massive mixed traffic," in *Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 4023–4029.

[73] H. Yu, W. Yang, R. Hao, C. Wang, J. Zhong, P. Luo *et al.*, "Drivee2e: Closed-loop benchmark for end-to-end autonomous driving through real-to-simulation," *arXiv preprint arXiv:2509.23922*, 2025. 8

[74] N. Moenne-Loccoz, A. Mirzaei, O. Perel, R. de Lutio, J. M. Esturo, G. State *et al.*, "3d gaussian ray tracing: Fast tracing of particle scenes," *ACM Trans. Graph. (TOG) and SIGGRAPH Asia*, 2024. 8

[75] A. Tonderski, C. Lindström, G. Hess, W. Ljungbergh, L. Svensson, and C. Petersson, "Neurad: Neural rendering for autonomous driving," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024, pp. 14 895–14 904.

[76] Q. Wu, J. Martinez Esturo, A. Mirzaei, N. Moenne-Loccoz, and Z. Gojcic, "3dgut: Enabling distorted cameras and secondary rays in gaussian splatting," *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025.

[77] Y. Chen, J. Zhang, Z. Xie, W. Li, F. Zhang, J. Lu *et al.*, "S-nerf++: Autonomous driving simulation via neural reconstruction and generation," *Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 2025.

[78] G. Hess, C. Lindström, M. Fatemi, C. Petersson, and L. Svensson, "Splatad: Real-time lidar and camera rendering with 3d gaussian splatting for autonomous driving," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025, pp. 11 982–11 992. 8

[79] NVIDIA, Y. Cao, R. de Lutio, S. Fidler, G. G. Cobo, Z. Gojcic *et al.*, "Alpasim: A modular, lightweight, and data-driven research simulator for autonomous driving," October 2025. [Online]. Available: https://github.com/NVlabs/alpasim 8, 9

[80] Z. Yang, Y. Chen, J. Wang, S. Manivasagam, W.-C. Ma, A. J. Yang *et al.*, "Unisim: A neural closed-loop sensor simulator," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 1389–1399. 8

[81] H. Zhou, L. Lin, J. Wang, Y. Lu, D. Bai, B. Liu *et al.*, "Hugsim: A real-time, photo-realistic and closed-loop simulator for autonomous driving," *arXiv preprint arXiv:2412.01718*, 2024. 8

[82] J. Z. Wu, Y. Zhang, H. Turki, X. Ren, J. Gao, M. Z. Shou *et al.*, "Difix3d+: Improving 3d reconstructions with single-step diffusion models," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025, pp. 26 024–26 035. 8, 9

[83] Z. Chen, J. Yang, J. Huang, R. de Lutio, J. M. Esturo, B. Ivanovic *et al.*, "OmniRe: omni urban scene reconstruction," in *Int. Conf. Learn. Represent. (ICLR)*, 2025. 8

[84] D. Charatan, S. Li, A. Tagliasacchi, and V. Sitzmann, "pixelSplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024. 8

[85] J. Yang, J. Huang, Y. Chen, Y. Wang, B. Li, Y. You *et al.*, "Storm: Spatio-temporal reconstruction model for large-scale outdoor scenes," *arXiv preprint arXiv:2501.00602*, 2024. 8

[86] Y. Hong, K. Zhang, J. Gu, S. Bi, Y. Zhou, D. Liu *et al.*, "LRM: large reconstruction model for single image to 3d," in *Int. Conf. Learn. Represent. (ICLR)*, 2024. 8

[87] H. Liang, J. Ren, A. Mirzaei, A. Torralba, Z. Liu, I. Gilitschenski *et al.*, "Feed-forward bullet-time reconstruction of dynamic scenes from monocular videos," *arXiv preprint arXiv:2412.03526*, 2024.

[88] L. Wang, S. W. Kim, J. Yang, C. Yu, B. Ivanovic, S. Waslander *et al.*, "Distillnerf: Perceiving 3d scenes from single-glance images by distilling neural fields and foundation model features," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 37, pp. 62 334–62 361, 2024.

[89] Z. Fan, J. Zhang, W. Cong, P. Wang, R. Li, K. Wen *et al.*, "Large spatial model: End-to-end unposed images to semantic 3d," in *Neural Inf. Process. Syst. (NeurIPS)*, 2024. 8

[90] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, 2000. 8

[91] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*, 2011. 8

[92] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Conf. Comput. Vis. Pattern Recognit. Worksh. (CVPRW)*, 2018, pp. 1468–1476. 8

[93] M. Igl, P. Shah, P. Mougin, S. Srinivasan, T. Gupta, B. White *et al.*, "Hierarchical imitation learning for stochastic environments," in *Int. Conf. Intell. Robots Syst. (IROS)*, 2023, pp. 1697–1704. 8

[94] Z. Zhou, J. Wang, Y.-H. Li, and Y.-K. Huang, "Query-centric trajectory prediction," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 17 863–17 873. 8

[95] R. Girgis, F. Golemo, F. Codevilla, M. Weiss, J. A. D'Souza, S. E. Kahou *et al.*, "Latent variable sequential set transformers for joint multi-agent motion prediction," *arXiv preprint arXiv:2104.00563*, 2021.

[96] J. Ngiam, B. Caine, V. Vasudevan, Z. Zhang, H.-T. L. Chiang, J. Ling *et al.*, "Scene transformer: A unified architecture for predicting multiple agent trajectories," *arXiv preprint arXiv:2106.08417*, 2021.

[97] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying," *Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 2024. 8

[98] E. Pronovost, M. R. Ganesina, N. Hendy, Z. Wang, A. Morales,

K. Wang *et al.*, "Scenario diffusion: Controllable driving scenario generation with diffusion," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, pp. 68 873–68 894, 2023. 8

[99] Z. Zhong, D. Rempe, D. Xu, Y. Chen, S. Veer, T. Che *et al.*, "Guided conditional diffusion for controllable traffic simulation," in *Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 3560–3566.

[100] C. Jiang, A. Cornman, C. Park, B. Sapp, Y. Zhou, D. Anguelov *et al.*, "Motiondiffuser: Controllable multi-agent motion prediction using diffusion," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 9644–9653.

[101] J. Lu, K. Wong, C. Zhang, S. Suo, and R. Urtasun, "Scenecontrol: Diffusion for controllable traffic scene generation," in *Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 16 908–16 914. 8

[102] M. Pei, S. Shi, and S. Shen, "Advancing multi-agent traffic simulation via r1-style reinforcement fine-tuning," *arXiv preprint arXiv:2509.23993*, 2025. 8

[103] N. Montali, J. Lambert, P. Mougin, A. Kuefler, N. Rhinehart, M. Li *et al.*, "The waymo open sim agents challenge," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, 2024. 8

[104] D. Rempe, J. Philion, L. J. Guibas, S. Fidler, and O. Litany, "Generating useful accident-prone driving scenarios via a learned traffic prior," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022. 8

[105] R. Gao, K. Chen, E. Xie, L. Hong, Z. Li, D.-Y. Yeung *et al.*, "Magicdrive: Street view generation with diverse 3d geometry control," *arXiv preprint arXiv:2310.02601*, 2023. 9

[106] S. Gao, J. Yang, L. Chen, K. Chitta, Y. Qiu, A. Geiger *et al.*, "Vista: A generalizable driving world model with high fidelity and versatile controllability," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 37, pp. 91 560–91 596, 2024. 9

[107] S. Tan, K. Wong, S. Wang, S. Manivasagam, M. Ren, and R. Urtasun, "Scenegen: Learning to generate realistic traffic scenes," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 892–901. 9

[108] L. Feng, Q. Li, Z. Peng, S. Tan, and B. Zhou, "Trafficgen: Learning to generate diverse and realistic traffic scenarios," in *Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 3567–3575. 9

[109] K. Chitta, D. Dauner, and A. Geiger, "Sledge: Synthesizing driving environments with generative models and rule-based traffic," in *Eur. Conf. Comput. Vis. (ECCV)*, 2024. 9

[110] L. Rowe, R. Girgis, A. Gosselin, L. Paull, C. Pal, and F. Heide, "Scenario dreamer: Vectorized latent diffusion for generating driving simulation environments," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025. 9

[111] S. Tan, B. Ivanovic, X. Weng, M. Pavone, and P. Kraehenbuehl, "Language conditioned traffic generation," in *Conf. Robot Learn. (CoRL)*, 2023. 9

[112] W. Ding, Y. Cao, D. Zhao, C. Xiao, and M. Pavone, "Realgen: Retrieval augmented generation for controllable traffic scenarios," in *Eur. Conf. Comput. Vis. (ECCV)*, 2024. 9

[113] W. Ding, C. Xu, M. Arief, H. Lin, B. Li, and D. Zhao, "A survey on safety-critical driving scenario generation—a methodological perspective," *Trans. Intell. Transp. Syst. (T-ITS)*, vol. 24, no. 7, pp. 6971–6988, 2023. 9

[114] Z. Zhong, D. Rempe, Y. Chen, B. Ivanovic, Y. Cao, D. Xu *et al.*, "Language-guided traffic simulation via scene-level diffusion," in *Conf. Robot Learn. (CoRL)*, 2023. 9

[115] S. Tan, B. Ivanovic, Y. Chen, B. Li, X. Weng, Y. Cao *et al.*, "Promptable closed-loop traffic simulation," in *Conf. Robot Learn. (CoRL)*, 2024. 9

[116] Z. Huang, Y. He, J. Yu, F. Zhang, C. Si, Y. Jiang *et al.*, "Vbench: Comprehensive benchmark suite for video generative models," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024, pp. 21 807–21 818. 9

[117] J. Bruce, M. D. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes *et al.*, "Genie: Generative interactive environments," in *Int. Conf. Mach. Learn. (ICML)*, 2024. 9

[118] P. J. Ball, J. Bauer, F. Belletti, B. Brownfield, A. Ephrat, S. Fruchter *et al.*, "Genie 3: A new frontier for world models," 2025. [Online]. Available: https://deepmind.google/discover/blog/genie-3-a-new-frontier-for-world-models/ 9

[119] A. Hu, L. Russell, H. Yeo, Z. Murez, G. Fedoseev, A. Kendall *et al.*, "Gaia-1: A generative world model for autonomous driving," *arXiv preprint arXiv:2309.17080*, 2023. 9

[120] X. Ren, Y. Lu, T. Cao, R. Gao, S. Huang, A. Sabour *et al.*, "Cosmos-drive-dreams: Scalable synthetic driving data generation with world foundation models," *arXiv preprint arXiv:2506.09042*, 2025. 9

[121] J. Yang, K. Chitta, S. Gao, L. Chen, Y. Shao, X. Jia *et al.*, "Resim: Reliable world simulation for autonomous driving," *arXiv preprint arXiv:2506.09981*, 2025. 9

[122] J. Wang, H. Sun, X. Yan, S. Feng, J. Gao, and H. X. Liu, "Terasim-world: Worldwide safety-critical data synthesis for end-to-end autonomous driving," *arXiv preprint arXiv:2509.13164*, 2025. 9

[123] Y. Zhou, M. Simon, Z. M. Peng, S. Mo, H. Zhu, M. Guo *et al.*, "Simgen: Simulator-conditioned driving scene generation," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 37, pp. 48 838–48 874, 2024. 9

[124] S. R. Richter, H. A. AlHaija, and V. Koltun, "Enhancing photorealism enhancement," *arXiv preprint arXiv:2105.04619*, 2021. 9

[125] Y. Lu, X. Ren, J. Yang, T. Shen, Z. Wu, J. Gao *et al.*, "Infinicube: Unbounded and controllable dynamic 3d driving scene generation with world-guided video models," *arXiv preprint arXiv:2412.03934*, 2024. 9

[126] D. Ha and J. Schmidhuber, "World models," *arXiv preprint arXiv:1803.10122*, vol. 2, no. 3, 2018. 9

[127] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104*, 2023. 9

[128] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "Daydreamer: World models for physical robot learning," in *Conf. Robot Learn. (CoRL)*, 2023, pp. 2226–2240. 9

[129] M. Assran, A. Bardes, D. Fan, Q. Garrido, R. Howes, M. Muckley *et al.*, "V-jepa 2: Self-supervised video models enable understanding, prediction and planning," *arXiv preprint arXiv:2506.09985*, 2025. 9

[130] G. Zhou, H. Pan, Y. LeCun, and L. Pinto, "Dino-wm: World models on pre-trained visual features enable zero-shot planning," *arXiv preprint arXiv:2411.04983*, 2024. 9, 10

[131] A. Hu, G. Corrado, N. Griffiths, Z. Murez, C. Gurau, H. Yeo *et al.*, "Model-based imitation learning for urban driving," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 35, pp. 20 703–20 716, 2022. 12

[132] A. Popov, A. Degirmenci, D. Wehr, S. Hegde, R. Oldja, A. Kamenev *et al.*, "Mitigating covariate shift in imitation learning for autonomous vehicles using latent space generative world models," *arXiv preprint arXiv:2409.16663*, 2024. 10, 12

[133] X. Wang, Z. Zhu, G. Huang, X. Chen, J. Zhu, and J. Lu, "Drivedreamer: Towards real-world-drive world models for autonomous driving," in *Eur. Conf. Comput. Vis. (ECCV)*, 2024, pp. 55–72.

[134] Y. Li, L. Fan, J. He, Y. Wang, Y. Chen, Z. Zhang *et al.*, "Enhancing end-to-end autonomous driving with latent world model," *arXiv preprint arXiv:2406.08481*, 2024. 12

[135] Q. Li, X. Jia, S. Wang, and J. Yan, "Think2drive: Efficient reinforcement learning by thinking in latent world model for quasi-realistic autonomous driving (in carla-v2)," *arXiv preprint arXiv:2402.16720*, 2024.

[136] Y. Zheng, P. Yang, Z. Xing, Q. Zhang, Y. Zheng, Y. Gao *et al.*, "World4drive: End-to-end autonomous driving via intention-aware physical latent world model," *arXiv preprint arXiv:2507.00603*, 2025. 9

[137] W. B. Knox, A. Allievi, H. Banzhaf, F. Schmitt, and P. Stone, "Reward (mis) design for autonomous driving," *Artif. Intell.*, vol. 316, p. 103829, 2023. 10

[138] A. Abouelazm, J. Michel, and J. M. Zöllner, "A review of reward functions for reinforcement learning in the context of autonomous driving," in *2024 IEEE Intelligent Vehicles Symposium (IV)*, 2024, pp. 156–163. 10

[139] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, "Dart: Noise injection for robust imitation learning," in *Conf. Robot Learn. (CoRL)*, 2017, pp. 143–156. 11

[140] M. Savari and Y. Choe, "Utilizing human feedback in autonomous driving: discrete vs. continuous," *Machines*, vol. 10, no. 8, p. 609, 2022. 11

[141] Z. Peng, W. Mo, C. Duan, Q. Li, and B. Zhou, "Learning from active human involvement through proxy value propagation," in *Neural Inf. Process. Syst. (NeurIPS)*, 2023. 11

[142] Z. Peng, Z. Liu, and B. Zhou, "Data-efficient learning from human interventions for mobile robots," in *Int. Conf. Robot. Autom. (ICRA)*, 2025. 11

[143] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end simulated driving," in *AAAI Conf. Artif. Intell.*, vol. 31, 2017. 11

[144] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Conf. Robot Learn. (CoRL)*, 2020, pp. 66–75.

[145] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool, "End-to-end urban driving by imitating a reinforcement learning coach," in *Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 15 222–15 232. 11, 13, 15

[146] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel *et al.*, "Stanley: The robot that won the DARPA Grand Challenge," *J. Field Robot. (JFR)*, vol. 23, no. 9, pp. 661–692, 2006. 11

[147] W. Pananurak, S. Thanok, and M. Parnichkun, "Adaptive cruise control for an intelligent vehicle," in *2008 IEEE International Conference on*

Robotics and Biomimetics, 2009, pp. 1794–1799. 11

[148] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *Int. Conf. Robot. Autom. (ICRA)*, 2010, pp. 987–993. 11

[149] G. Garcia-Cobo, M. Igl, P. Karkus, Z. Zhang, M. Watson, Y. Chen *et al.*, "Road: Rollouts as demonstrations for closed-loop supervised fine-tuning of autonomous driving policies," *arXiv preprint arXiv:2512.01993*, 2025. 11

[150] L. Cultrera, F. Becattini, L. Seidenari, P. Pala, and A. D. Bimbo, "Addressing limitations of state-aware imitation learning for autonomous driving," *Trans. Intell. Veh. (T-IV)*, vol. 9, no. 1, pp. 2946–2955, 2024. 11

[151] M. Vitelli, Y. Chang, Y. Ye, A. Ferreira, M. Wołczyk, B. Osiński *et al.*, "Safetynet: Safe planning for real-world self-driving vehicles using machine-learned policies," in *Int. Conf. Robot. Autom. (ICRA)*, 2022, pp. 897–904. 11

[152] J. Cheng, Y. Chen, X. Mei, B. Yang, B. Li, and M. Liu, "Rethinking imitation-based planners for autonomous driving," in *Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 14 123–14 130. 11

[153] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015. 12

[154] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 1889–1897. 12

[155] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015. 12

[156] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 1587–1596. 12

[157] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015. 12

[158] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992. 12

[159] Z. Peng, W. Luo, Y. Lu, T. Shen, C. Gulino, A. Seff *et al.*, "Improving agent behaviors with rl fine-tuning for autonomous driving," in *Eur. Conf. Comput. Vis. (ECCV)*, 2025, pp. 165–181. 12, 14

[160] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, pp. 53 728–53 741, 2023. 13, 14

[161] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025. 13

[162] C. Zheng, S. Liu, M. Li, X.-H. Chen, B. Yu, C. Gao *et al.*, "Group sequence policy optimization," *arXiv preprint arXiv:2507.18071*, 2025. 13

[163] Z. Liu, C. Chen, W. Li, P. Qi, T. Pang, C. Du *et al.*, "Understanding r1-zero-like training: A critical perspective," *arXiv preprint arXiv:2503.20783*, 2025.

[164] Z. Liu, J. Liu, Y. He, W. Wang, J. Liu, L. Pan *et al.*, "Part i: Tricks or traps? a deep dive into rl for llm reasoning," *arXiv preprint arXiv:2508.08221*, 2025. 13

[165] B. Jiang, S. Chen, Q. Zhang, W. Liu, and X. Wang, "Alphadrive: Unleashing the power of vlms in autonomous driving via reinforcement learning and reasoning," *arXiv preprint arXiv:2503.07608*, 2025. 13

[166] Y. Li, M. Tian, D. Zhu, J. Zhu, Z. Lin, Z. Xiong *et al.*, "Drive-r1: Bridging reasoning and planning in vlms for autonomous driving with reinforcement learning," *arXiv preprint arXiv:2506.18234*, 2025.

[167] R. Tian and K. Goel, "Direct post-training preference alignment for multi-agent motion generation models using implicit feedback from pre-training demonstrations," *arXiv preprint arXiv:2503.20105*, 2025. 13

[168] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," in *Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 2034–2039. 13

[169] D. M. Saxena, S. Bae, A. Nakhaei, K. Fujimura, and M. Likhachev, "Driving in dense traffic with model-free reinforcement learning," in *Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 5385–5392.

[170] M. Harmel, A. Paras, A. Pasternak, N. Roy, and G. Linscott, "Scaling is all you need: Autonomous driving with jax-accelerated reinforcement learning," *arXiv preprint arXiv:2312.15122*, 2023. 13

[171] M. U. Yavas, T. Kumbasar, and N. K. Ure, "A real-world reinforcement learning framework for safe and human-like tactical decision-making," *Trans. Intell. Transp. Syst. (T-ITS)*, vol. 24, no. 11, pp. 11 773–11 784, 2023.

[172] B. Jaeger, D. Dauner, J. Beißwenger, S. Gerstenecker, K. Chitta, and A. Geiger, "Carl: Learning scalable planning policies with simple rewards," *arXiv preprint arXiv:2504.17838*, 2025. 13

[173] J. Chen, B. Yuan, and M. Tomizuka, "Model-free deep reinforcement learning for urban autonomous driving," in *Int. Conf. Intell. Transp. Syst. (ITSC)*, 2019, pp. 2765–2771. 13

[174] M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," in *Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 7153–7162.

[175] R. Chekroun, M. Toromanoff, S. Hornauer, and F. Moutarde, "Gri: General reinforced imitation and its application to vision-based autonomous driving," *Robotics*, vol. 12, no. 5, p. 127, 2023. 13

[176] R. Tian, S. Li, N. Li, I. Kolmanovsky, A. Girard, and Y. Yildiz, "Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts," in *Conf. Dec. and Cont. (CDC)*, 2018, pp. 321–326. 13

[177] C.-J. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, and M. J. Kochenderfer, "Combining planning and deep reinforcement learning in tactical decision making for autonomous driving," *Trans. Intell. Veh. (T-IV)*, vol. 5, no. 2, pp. 294–305, 2019.

[178] X. Ma, J. Li, M. J. Kochenderfer, D. Isele, and K. Fujimura, "Reinforcement learning for autonomous driving with latent state inference and spatial-temporal relationships," in *Int. Conf. Robot. Autom. (ICRA)*, 2021, pp. 6064–6071. 13

[179] A. Pan, K. Bhatia, and J. Steinhardt, "The effects of reward misspecification: Mapping and mitigating misaligned models," *arXiv preprint arXiv:2201.03544*, 2022. 13

[180] Z. Li, K. Li, S. Wang, S. Lan, Z. Yu, Y. Ji *et al.*, "Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation," *arXiv preprint arXiv:2406.06978*, 2024. 13

[181] Z. Zhou, T. Cai, S. Z. Zhao, Y. Zhang, Z. Huang, B. Zhou *et al.*, "Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning," *arXiv preprint arXiv:2506.13757*, 2025. 13

[182] F. Bartoccioni, E. Ramzi, V. Besnier, S. Venkataramanan, T.-H. Vu, Y. Xu *et al.*, "Vavim and vavam: Autonomous driving through video generative modeling," *arXiv preprint arXiv:2502.15672*, 2025. 13

[183] J. Liu, G. Liu, J. Liang, Y. Li, J. Liu, X. Wang *et al.*, "Flow-grpo: Training flow matching models via online rl," *arXiv preprint arXiv:2505.05470*, 2025. 13

[184] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," *arXiv preprint arXiv:1710.06542*, 2017. 13

[185] P. J. Ball, L. Smith, I. Kostrikov, and S. Levine, "Efficient online reinforcement learning with offline data," in *Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 1577–1594. 13, 14

[186] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *AAAI Conf. Artif. Intell.*, vol. 8, 2008, pp. 1433–1438. 14

[187] S. Sharifzadeh, I. Chiotellis, R. Triebel, and D. Cremers, "Learning to drive using inverse reinforcement learning and deep q-networks," *arXiv preprint arXiv:1612.03653*, 2016. 14

[188] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions." in *Robot.: Sci. Syst. (RSS)*, vol. 2, 2016, pp. 1–9. 14

[189] R. Tian, M. Tomizuka, and L. Sun, "Learning human rewards by inferring their latent intelligence levels in multi-agent games: A theory-of-mind approach with application to driving data," in *Int. Conf. Intell. Robots Syst. (IROS)*, 2021, pp. 4560–4567.

[190] Z. Huang, H. Liu, J. Wu, and C. Lv, "Conditional predictive behavior planning with inverse reinforcement learning for human-like autonomous driving," *Trans. Intell. Transp. Syst. (T-ITS)*, vol. 24, no. 7, pp. 7244–7258, 2023. 14

[191] Z. Huang, Z. Sheng, Y. Qu, J. You, and S. Chen, "Vlm-rl: A unified vision language models and reinforcement learning framework for safe autonomous driving," *arXiv preprint arXiv:2412.15544*, 2024. 14

[192] A. Jiang, Y. Gao, Y. Wang, Z. Sun, S. Wang, Y. Heng *et al.*, "Irl-vla: Training an vision-language-action policy via reward world model," *arXiv preprint arXiv:2508.06571*, 2025. 14

[193] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan, "Inverse reward design," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017. 14

[194] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," in *Int. Conf. Learn. Represent. (ICLR)*, 2018. 14

[195] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017. 14

[196] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin *et al.*, "Training language models to follow instructions with human feedback," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 35, pp. 27 730–27 744, 2022. 14

[197] R. Tian, Y. Wu, C. Xu, M. Tomizuka, J. Malik, and A. Bajcsy, "Maximizing alignment with minimal feedback: Efficiently learning rewards for visuomotor robot policy alignment," *arXiv preprint arXiv:2412.04835*, 2024. 14

[198] H. Li, W. Xu, J. Sun *et al.*, "Trajhf: Trajectory-level reinforcement learning from human feedback for autonomous driving," in *Int. Conf. Learn. Represent. (ICLR)*, 2025.

[199] S. Shang, Y. Chen, Y. Wang, Y. Li, and Z. Zhang, "Drivedpo: Policy learning via safety dpo for end-to-end autonomous driving," *arXiv preprint arXiv:2509.17940*, 2025. 14

[200] A. R. Srinivasan, J. Schumann, Y. Wang, Y.-S. Lin, M. Daly, A. Solernou *et al.*, "The commotions urban interactions driving simulator study dataset," *arXiv preprint arXiv:2305.11909*, 2023. 14

[201] X. Li, A. Rakotonirainy, and X. Yan, "How do drivers avoid collisions? a driving simulator-based study," *Journal of safety research*, vol. 70, pp. 89–96, 2019. 14

[202] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 29, 2016. 14

[203] Y. Li, J. Song, and S. Ermon, "Infogail: Interpretable imitation learning from visual demonstrations," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017. 14

[204] F. Torabi, G. Warnell, and P. Stone, "Generative adversarial imitation from observation," *arXiv preprint arXiv:1807.06158*, 2018.

[205] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson, "Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning," *arXiv preprint arXiv:1809.02925*, 2018.

[206] I. Kostrikov, O. Nachum, and J. Tompson, "Imitation learning via off-policy distribution matching," *arXiv preprint arXiv:1912.05032*, 2019. 14

[207] G. C. K. Couto and E. A. Antonelo, "Generative adversarial imitation learning for end-to-end autonomous driving on urban environments," in *Symp. Ser. Comput. Intell. (SSCI)*, 2021, pp. 1–7. 14

[208] E. Bronstein, M. Palatucci, D. Notz, B. White, A. Kuefler, Y. Lu *et al.*, "Hierarchical model-based imitation learning for planning in autonomous driving," *arXiv preprint arXiv:2210.09539*, 2022.

[209] G. C. K. Couto and E. A. Antonelo, "Hierarchical generative adversarial imitation learning with mid-level input generation for autonomous driving on urban environments," *Trans. Intell. Veh. (T-IV)*, 2024. 14

[210] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020. 14

[211] H. Van Hasselt, Y. Doron, F. Strub, M. Hessel, N. Sonnerat, and J. Modayil, "Deep reinforcement learning and the deadly triad," *arXiv preprint arXiv:1812.02648*, 2018. 14

[212] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," in *Neural Inf. Process. Syst. (NeurIPS)*, 2020. 14

[213] X. B. Peng, A. Kumar, G. Zhang, and S. Levine, "Advantage-weighted regression: Simple and scalable off-policy reinforcement learning," *arXiv preprint arXiv:1910.00177*, 2019. 14

[214] Z. Wang, A. Novikov, K. Zolna, J. S. Merel, J. T. Springenberg, S. E. Reed *et al.*, "Critic regularized regression," *Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, pp. 7768–7778, 2020. 14

[215] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *AAAI Conf. Artif. Intell.*, vol. 30, 2016. 14

[216] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot *et al.*, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017. 14

[217] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov *et al.*, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017. 14

[218] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot *et al.*, "Deep q-learning from demonstrations," in *AAAI Conf. Artif. Intell.*, vol. 32, 2018. 14

[219] A. Nair, A. Kumar, C. Finn, and S. Levine, "Accelerating online reinforcement learning with offline datasets," in *Conf. Robot Learn. (CoRL)*, 2020. 14

[220] Y. Lu, J. Fu, G. Tucker, X. Pan, E. Bronstein, R. Roelofs *et al.*, "Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios," in *Int. Conf. Intell. Robots Syst. (IROS)*, 2023, pp. 7553–7560. 14

[221] A. Kamenev, L. Wang, O. B. Bohan, I. Kulkarni, B. Kartal, A. Molchanov *et al.*, "Predictionnet: Real-time joint probabilistic traffic prediction for planning, control, and simulation," in *Int. Conf. Robot. Autom. (ICRA)*, 2022, pp. 8936–8942. 14

[222] Z. Huang, J. Wu, and C. Lv, "Efficient deep reinforcement learning with imitative expert priors for autonomous driving," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7391–7403, 2022. 14

[223] Z. Cao, E. Bıyık, W. Z. Wang, A. Raventos, A. Gaidon, G. Rosman *et al.*, "Reinforcement learning based control of imitative policies for near-accident driving," *arXiv preprint arXiv:2007.00178*, 2020. 14

[224] L. Wang, J. Liu, H. Shao, W. Wang, R. Chen, Y. Liu *et al.*, "Efficient reinforcement learning for autonomous driving with parameterized skills and priors," *arXiv preprint arXiv:2305.04412*, 2023. 15

[225] K. Li, Z. Li, S. Lan, Y. Xie, Z. Zhang, J. Liu *et al.*, "Hydra-mdp++: Advancing end-to-end driving via expert-guided hydra-distillation," *arXiv preprint arXiv:2503.12820*, 2025. 15

[226] M. Cusumano-Towner, D. Hafner, A. Hertzberg, B. Huval, A. Petrenko, E. Vinitsky *et al.*, "Robust autonomy emerges from self-play," *arXiv preprint arXiv:2502.03349*, 2025. 15

[227] Q. Yin, T. Yu, S. Shen, J. Yang, M. Zhao, W. Ni *et al.*, "Distributed deep reinforcement learning: A survey and a multi-player multi-agent learning toolbox," *Machine Intelligence Research*, vol. 21, no. 3, pp. 411–430, 2024. 15

[228] Z. Liu, X. Xu, P. Qiao, and D. Li, "Acceleration for deep reinforcement learning using parallel and distributed computing: A survey," *ACM Computing Surveys*, vol. 57, no. 4, pp. 1–35, 2024. 15

Author Biographies

Peter Karkus received the PhD degree from the National University of Singapore, Singapore. He is a Senior Research Scientist at NVIDIA.

Maximilian Igl received the PhD degree from the University of Oxford, UK. He is a Senior Research Scientist at NVIDIA.

Yuxiao Chen received the PhD degree from the University of Michigan, USA. He is a Senior Research Scientist at NVIDIA.

Kashyap Chitta received his PhD degree from the University of Tübingen, Germany. He is a Postdoctoral Scholar at NVIDIA.

Jef Packer received his MEng degree from University of California, Davis. He is a Senior Software Engineer at NVIDIA.

Bertrand Douillard received his PhD from the University of Sydney, Australia. He is a Distinguished Research Engineer at NVIDIA.

Ran Tian received his PhD degree from UC Berkeley, USA. He is a Research Scientist at NVIDIA.

Alexander Naumann received his PhD degree from the Karlsruhe Institute of Technology, Germany. He is an Applied Scientist at NVIDIA.

Guillermo Garcia-Cobo received his MS degree from Columbia University, NY, USA. He is a Software Engineer at NVIDIA.

Shuhan Tan received his BA degree from Sun Yat-sen Univeristy, China. He is a Research Intern at NVIDIA.

Alperen Degirmenci received his PhD degree from Harvard University, MA, USA. He is a Senior Applied Research Scientist at NVIDIA.

Alexander Popov received his MS degree from the University of Minnesota, MN, USA. He is a Senior Applied Researcher at NVIDIA.

Nikolai Smolyanskiy received his PhD in Applied Math from South Ural State University, Russia. He is a Director in the Autonomous Vehicles division at NVIDIA.

Urs Muller received his PhD degree from ETH Zurich, Switzerland. He is a Senior Director in the Autonomous Vehicles division at NVIDIA.

Boris Ivanovic received the PhD degree from Stanford University, USA. He is a Senior Research Scientist and Manager at NVIDIA.

Marco Pavone received the PhD degree from the Massachusetts Institute of Technology, USA. He is a Senior Director of Autonomous Driving Research at NVIDIA and Associate Professor in the Department of Aeronautics and Astronautics at Stanford University.