Human-like autonomy emerges from self-play and a pinch of human data

Anonymous Author(s)
Affiliation
Address
email

Abstract: Self-play reinforcement learning has recently taken the wheel in autonomous driving. The approach uses cheap, large-scale simulation to substitute expensive, large-scale human driving data used by imitation learning approaches. While cost-effective, policies trained through self-play often behave unpredictably around humans; they fail to coordinate because they tend to converge to effective but incompatible driving conventions. Previous works attempt to mitigate such behavioral incompatibilities through reward engineering and domain randomization techniques, which are brittle and labor-intensive. We build on the best of both approaches and instead train driving policies using 30 minutes of human driving data and over 60 years of self-play simulation with a minimal reward function that solely encodes safe goal reaching. We call this *spiced self-play*: just as a pinch of a well-chosen ingredient can transform a dish, a small amount of human data, added as a behavioral anchor, disproportionately improves coordination with human drivers. Results show that our policies achieve improved coordination with real human driving trajectories using 2,500× less human data than imitation learning baselines. The entire pipeline completes in 15 hours on a single consumer-grade GPU. Together, our results point toward a positive revision of the field's understanding of data scaling: a small quantity of human data may suffice to regularize the vast space of driving policies toward ones that fit the road.

Keywords: Self-play Reinforcement Learning, Imitation Learning, Autonomous Driving

1 Introduction

Self-play reinforcement learning (RL) has produced superhuman agents in strategic games [1, 2, 3] and, more recently, has shown promise in real-world domains, including autonomous driving [4, 5, 6, 7] and robotic manipulation [8]. The approach elegantly sidesteps a central difficulty in multi-agent learning - how to model the opponent - through the following idea: the *agent's opponent is a copy of itself*. The appeal here is that as the agent improves, so does its co-player. This gives rise to an automatically evolving curriculum [9] that takes behavior from random play to skilled behavior entirely through synthetic simulated experience.

In zero-sum games, this mechanism, with a sparse measure for success (e.g., +1 when winning a game of chess), is enough to produce strong play against arbitrary opponents. Many real-world settings, however, are not zero-sum. Driving, for instance, can be viewed as a mixed-motive game: each player has *individual objectives* (reaching a destination safely) but must also *coordinate* with other road users by adhering to shared norms, expectations, and conventions. Self-play RL with only a high-level objective for success provides no guarantees of such alignment; policies may converge to effective but "alien" strategies that are incompatible with human partners [10]. Concretely, an agent trained to "reach a destination safely" may very well learn to do so in reverse, sideways, or on the wrong side of the road if such constraints are not specified in the reward.

Submitted to the 10th Conference on Robot Learning (CoRL 2026). Do not distribute.

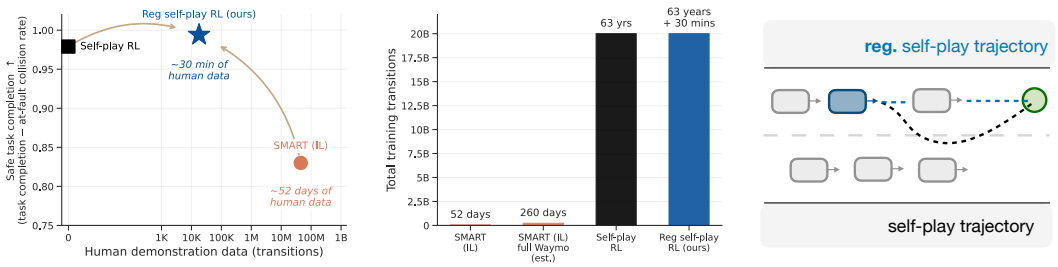

Figure 1: **Spiced self-play RL achieves human-like coordination from 30 minutes of human data and 60 years of simulated experience.** *Left:* Safe task completion (task completion rate − at-fault collision rate) against human driving data, evaluated against human-replay proxies. With ∼30 min of human driving data as a behavioral anchor (★, ours; 0.994), our method outperforms unregularized self-play (■; 0.979) and SMART-tiny CLSFT [11] (●; 0.830), an IL-based approach trained on the full Waymo dataset. Beige arrows show improvement over each baseline. *Center:* Total training transitions used per method. Both self-play variants consume 20B transitions (∼63 years at 10 Hz) of cheap synthetic experience; SMART uses 45M–225M human logged transitions (∼52 days–7 months; see Appendix G). *Right:* Example rollouts. The self-play policy (- - -) drives aggressively and threads the needle when there are gaps; the regularized policy (- - -) yields politely to other agents. Dark-blue vehicle is the controlled agent, which is goal-conditioned on the green target destination. Grey agents follow log replay.

Previous works have addressed this misalignment in two ways. One line of work involves *manual reward engineering*, where reward terms are added iteratively until the desired behavior and conventions emerge [5, 12]. While effective, this strategy is labor-intensive by nature, domain-specific, and brittle since it is not trivial to figure out what reward will produce the desired human-like behavior [13]. A case in point is GIGAFLOW [5], which required nine individually tuned reward terms and several other domain randomization tricks to produce naturalistic and cautious driving policies. On the other side of the spectrum, we have *Imitation Learning* [14, 15, 16, IL]. IL directly imitates human driving data to avoid having to define a reward function altogether. However, robustness requires wide state coverage, so these approaches typically need large quantities of human data to work well [17].

We take a different approach, grounded in a practical observation about the changing cost structure of experience generation. Modern RL frameworks and simulation infrastructure can generate between 300K and 20M environment steps *per second* on a single consumer-grade GPU [18, 19], making synthetic experience generation effectively limitless. Human driving data, by contrast, requires manual collection and remains slow to scale. This suggests a natural role for human data in coordination games: not as the primary source of training signal, but as a lightweight anchor that steers the policy away from effective yet behaviorally alien strategies. Indeed, regularizing self-play RL toward such an anchor has shown promise in producing human-compatible agents in Diplomacy [20, 21] and driving [22, 23, 7], yet *how much* data is required to reach human compatibility remains, to our knowledge, unexamined.

We measure it. Anchoring self-play RL to human driving data from the Waymo Open Motion Dataset [24, WOMD], we find that a surprisingly small amount of human demonstration data improves coordination with human proxies. Together with roughly 60 years of self-play experience, 30 minutes of human driving data (0.04% of the full WOMD train dataset) produces a marked improvement, without any reward engineering or domain randomization. Inspired by the disproportion between these two data sources, we name our method *spiced self-play*. Just as a pinch of cayenne is enough to change the flavor of an entire dish, a small amount of human data can substantially reshape the behavior of a self-play policy. Concretely, we train a PPO policy [25] under a sparse reward for safe goal reaching, while *regularizing* it toward a behavioral cloning anchor fit to a small amount of human driving data. Our contributions are:

- We demonstrate that 30 minutes to 3 hours of human driving data, combined with self-play at scale, is sufficient to improve coordination with human proxies without reward engineering or domain randomization (Figure 1; Section 4.1).
- We find that spiced policies not only have lower collision rates, they also display more human-like behavior in terms of distributional realism [26] and collision severity profiles [27] (Section 4.2).
- To make it easy to reproduce and build on the current results, we open-source the full codebase. Policies can be trained end-to-end in 15 hours on a single consumer-class GPU.

2 Related Work

Imitation learning for autonomous driving. The generation of driving policies is a fundamental challenge across end-to-end autonomous driving [28, 29, 30, 31], multi-agent trajectory prediction [32], and reactive traffic simulation [26, 33]. Driven by the widespread availability of large-scale human driving datasets [34, 24, 35], imitation learning has become the dominant approach across all these domains [36]. Under this imitation learning paradigm, a broad spectrum of methodologies has emerged to fit models to historical data, ranging from marginal [37, 38, 39] and joint [40, 41, 42, 43] forecasting to autoregressive sequence modeling of tokenized trajectories [44, 16, 11] and continuous distribution learning via diffusion and promptable world models [45, 46, 47, 48, 49]. While these generative approaches yield diverse open-loop behaviors, they are fundamentally constrained by the scale of human data required and frequently suffer from compounding covariate shift in closed-loop deployment [17]. To mitigate these shifts, recent hybrid approaches integrate reinforcement learning [50, 51, 52], yet they typically still rely on extensive human driving data as their primary optimization signal. Our approach systematically inverts this balance: rather than depending on human driving data as the core supervisor, we utilize synthetic, multi-agent RL self-play as the primary engine for discovering robust interactive behaviors, retaining a remarkably small human dataset strictly as a behavioral anchor to ensure conformity to realistic traffic norms.

Self-play reinforcement learning in games. Self-play reinforcement learning has produced superhuman agents in games from Go and Chess [1, 53] to StarCraft II [54] and Stratego [2], all without human data. Superhuman play is not the same as human-compatible play, however. Many games admit multiple equilibria, and self-play need not converge to equilibria that are compatible with human partners [10, 55]. The failure has been shown in cooperative games such as Hanabi [56] and Diplomacy [10], where self-play agents develop internally consistent conventions that transfer poorly to human partners. The cause is reward underspecification: when the reward is defined as a score to maximize, there are often many ways to achieve it. In other words, the solution space is large. Previous work attempts to resolve this by sharing the reward by hand [5, 12]. For instance, GIGAFLOW [5] demonstrates that reward engineering and domain randomization can produce naturalistic behavior at scale, at the cost of nine individually tuned reward terms. We avoid reward engineering entirely. A small amount of human data serves as a behavioral anchor, and self-play does the rest. This reduces a labor-intensive design problem to a one-hour data collection.

Human-regularized self-play reinforcement learning and search. One alternative to reward engineering is to regularize self-play toward a human anchor policy. This idea has been explored in Diplomacy, where KL regularization toward a human prior during both search and learning produced agents that coordinate more effectively with human partners [20, 21]. Jacob et al. [57] study KL-regularized search more broadly and show that it recovers human-like play across several games. In autonomous driving, the idea has been explored at a limited scale [22, 7]. Chang et al. [7] demonstrates that KL-regularized self-play can yield human-like driving policies using SMART [11] as the behavioral anchor, a large tokenized model trained on the full 500,000-scenario Waymo dataset. However, their setup allows only 1 billion self-play steps and replays vulnerable road users (VRUs; pedestrians and cyclists) agents from human data during training, which conflates the anchor's contribution with that of the mixed-in human trajectories and precludes a clean analysis of where the impact comes from. We scale self-play to 20 billion steps, control all agents during self-play

training to preclude human contamination of collected human data, and systematically study how much human anchor data is needed to improve human compatibility

3 Method

3.1 Evaluation and Metrics

Problem setup. A human-compatible agent should *blend in* with human drivers. Since on-road deployment is out of scope, we approximate interaction with human road users by replaying logged human trajectories in simulation. We evaluate in three settings, illustrated in Figure 11:

- **Self-play.** All agents are controlled by the **same policy** in a decentralized manner.
- **Human-replay.** Only the self-driving car (SDC) is controlled by the policy; all other agents follow their logged trajectories.
- **IDM.** The SDC is controlled by the policy; all other agents follow the Intelligent Driver Model [58], following a precomputed lane-center path for lateral control and using longitudinal accelerations of IDM to maintain a safe gap between the lead vehicle [59].

An effective and human-compatible agent should reach its goal without collisions or off-road events across all three settings, each of which probes a distinct failure mode. Human-replay tests whether the policy has internalized human driving conventions against non-reactive co-players. IDM introduces closed-loop dynamics with reactive rule-based co-players. Self-play tests internal consistency and additionally serves as a convergence sanity check.

Metrics. We report several metrics that capture task performance. The **score** is an aggregate metric; an agent scores 1 if it completes the task of driving to a goal destination before the end of the episode without colliding or going off-road, and 0 otherwise. To diagnose failure modes, we separately report **collision rate**, **at-fault collision rate**, **off-road rate**, and **route progress**. An ideal agent should score well with its own population as well as the human-replay population. Score-based metrics capture whether agents complete their task safely, but not whether their behavior looks human. We therefore also report **distributional realism** using the Waymo Open Sim Agent Challenge [26] to compare their behavior to logged trajectories. Finally, we also analyze the severity of the at-fault collisions [27]. Metrics are reported on **held-out test scenarios** unless stated otherwise; see full definitions and details in Appendix D.3.

3.2 Simulation Environment

World initialization. We use PufferDrive 2.0 [19] for simulation and training. Environments are initialized from the Waymo Open Motion Dataset [24, WOMD]: each 9-second scenario provides a roadgraph, a variable set of agents (cars, cyclists, pedestrians) up to $N = 32$, and per-agent initial poses and goals drawn from the logs. Each agent is goal-conditioned on a target destination (x, y position) and receives a partial, decentralized, ego-frame observation consisting of its own state, the $N - 1$ closest neighbors within 50 m, and up to 128 nearby road segments. World initialization and observation space details are provided in Appendix A.1 and A.2, respectively.

Reward function. To isolate the effect of human driving data, we avoid reward engineering and use a sparse reward: $+1$ for reaching the goal, -1 for collision or off-road events, and 0 otherwise. Any differences in human-like behavior, therefore, stem from BC regularization rather than a hand-tuned reward. Episodes terminate once all agents reach their destinations, and we filter out transitions from agents that reach their goals early.

3.3 Spiced Self-Play Reinforcement Learning

Spiced self-play is *regularized* self-play RL anchored to a small amount of human demonstration data (here driving logs). The anchor is a behavioral cloning policy fit to this data, which regularizes

self-play through a KL penalty. We train policies in two stages: a behavioral cloning (BC) anchor is first fit to human data, then frozen and used as a regularizer during self-play RL.

Step 1: Train the anchor policy. To study how the amount of human data affects downstream performance, we train anchor policies on subsets of the full dataset $\mathcal{D} = \{(o_t^i, a_t^i)\}_{i=1}^{T \cdot K}$. We sample subsets $\mathcal{D}_n$ corresponding to n scenarios, yielding roughly $\{10\text{ min}, 30\text{ min}, 3\text{ h}, 30\text{ h}\}$ of human driving data, and train each anchor by minimizing negative log-likelihood:

$$\tau_\phi^n = \arg\min_\phi \sum_{(o_t^i, a_t^i) \in \mathcal{D}_n} -\log \tau_\phi^n(a_t^i \mid o_t^i). \tag{1}$$

We use only the SDC trajectory from each scenario to generate our imitation data, as it is typically the highest-quality trajectory. Each τ_ϕ^n is then frozen for the subsequent self-play stage. Full details are in Appendix A.4.

Step 2: Regularized self-play RL. We train π_θ from scratch using Proximal Policy Optimization [25, PPO]. The policy π_θ is represented by a 650k-parameter neural network. Each anchor τ_ϕ^n serves as a behavioral regularizer via a KL penalty:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{PPO}}(\theta) + \lambda\, \mathbb{E}_{o \sim \rho_{\pi_\theta}} \left[D_{\text{KL}}\Big(\tau_\phi^n(\cdot \mid o) \,\Big\|\, \pi_\theta(\cdot \mid o) \Big) \right], \tag{2}$$

where ρ_{π_θ} is the on-policy state distribution and $\lambda \geq 0$ controls regularization strength. The KL term pulls π_θ toward the anchor on states the policy actually visits, rather than on the offline distribution of $\mathcal{D}_n$. Hyperparameters and training details are in Appendices A.1 and B.

4 Experiments

This section summarizes the key results. Additional details and analyses are reported in the appendices. We structure the sections to answer the following questions:

1. *Scaling human driving data for regularized self-play RL*: How much human data is needed for strong performance in both self-play and human-replay evaluations? (Section 4.1)
2. *Behavior and safety analysis*: How does a small amount of human demonstration data shape policy behavior beyond task performance? We analyze the effect on distributional realism, collision severity, and driving style (Section 4.2).
3. *The role of metadata and scenario diversity*: Driving datasets such as WOMD and NuPlan provide *scenario metadata*—road graphs and initial agent positions—that ground simulation at a fraction of the cost of collecting human driving data. How does the number of training scenarios (maps) used for self-play influence agent performance? (Section 4.3)

4.1 Scaling Human Driving Data for Regularized Self-Play RL

How much collected human driving data does regularized self-play need, and how does this compare to imitation learning-based approaches? The second question matters because any apparent data efficiency on our side could simply reflect the homogeneity of the Waymo Open Dataset rather than an actual property of the method. We benchmark against vanilla self-play RL (**- - -**), which provides a human-data-free lower bound, and SMART-tiny-CLSFT [11, 52] (—●—), the state-of-the-art IL approach in this domain. SMART is trained on the same nested driving data subsets; we additionally include the open-sourced SMART-tiny-CATK checkpoint [52], trained on all 500k WOMD training scenarios, as an IL upper bound (Appendix B.3).

Regularized self-play RL surpasses IL with a fraction of the human driving data. As shown in Figure 2 and Table 1, regularized self-play outperforms SMART-tiny-CLSFT across all data regimes and metrics. With as little as 30 minutes to 3 hours of human data, regularized self-play achieves the lowest at-fault collision rate (0.6%); a 2.5× improvement over SMART-tiny-CLSFT trained on 52 days of data (1.6%). The advantage is most pronounced at low human data: at 30 minutes, regularized

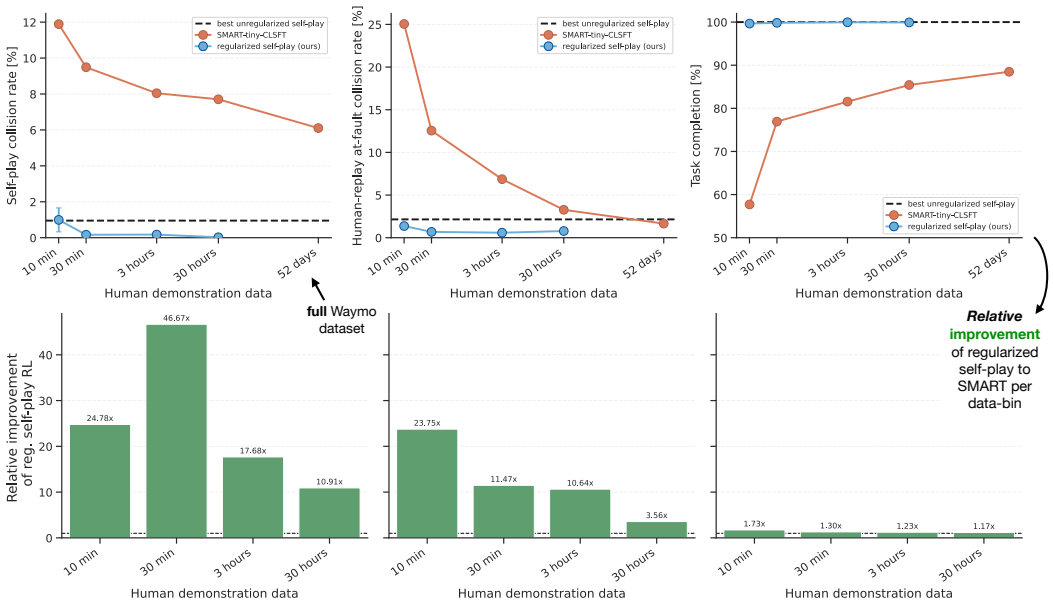

Figure 2: **Scaling human driving data for spiced self-play reinforcement learning.** *Top*: Performance of *regularized* self-play RL (—●—) and SMART with CAT-K closed-loop fine-tuning (—●—) as a function of total human log data used for training, evaluated in self-play and against human replays. Policies are evaluated on the same random 10k held-out WOMD validation split [24]. Vanilla self-play RL (- - -) is shown as a horizontal line, since it uses no human driving data. The horizontal axis is semi-logarithmic. *Bottom*: Relative improvement to IL baseline.

self-play yields an 11× reduction in at-fault collision rate and 46× in self-play collision rate relative to SMART. Against vanilla self-play RL (at-fault CR: 2.1%; - - -), regularized self-play achieves a 3.5× improvement, demonstrating the value of an anchor trained on minimal human data as a regularizer.

Table 1: Performance vs. amount of human driving data for the best trained policies on 10,000 held out randomly sampled scenarios. Top-3 values per column are highlighted (best , 2nd , 3rd); best value additionally in bold. The unregularized self-play row uses no human driving data.

Human demos used	Method	Self-play (test)			Human-replay (test)				
		Coll. (%) ↓	Off-road (%) ↓	Route prog. (%) ↑	Score ↑	Coll. (%) ↓	At-fault (%) ↓	Off-road (%) ↓	Route prog. (%) ↑
10 min	SMART	11.9	55.8	84.5	0.246	32.0	25.0	18.6	57.7
30 min	SMART	9.5	55.4	85.8	0.379	17.9	12.5	16.8	76.9
3 hours	SMART	8.0	53.6	86.2	0.518	11.4	6.9	4.5	81.5
52 days	SMART	6.1	53.5	91.7	0.654	4.4	1.6	1.1	88.5
—	unreg. self-play	1.0 ± 0.4	**0.2 ± 0.2**	**99.9 ± 0.1**	0.967 ± 0.006	2.7 ± 0.5	2.1 ± 0.5	**0.6 ± 0.2**	**100.0 ± 0.0**
10 min	reg. self-play (ours)	1.0 ± 0.7	0.3 ± 0.2	99.0 ± 0.4	0.941 ± 0.007	3.9 ± 0.6	1.4 ± 0.4	1.4 ± 0.4	99.6 ± 0.2
30 min	reg. self-play (ours)	0.2 ± 0.1	0.5 ± 0.2	99.3 ± 0.3	0.968 ± 0.006	2.0 ± 0.4	0.7 ± 0.3	1.4 ± 0.4	99.8 ± 0.1
3 hours	reg. self-play (ours)	0.2 ± 0.1	0.6 ± 0.4	99.6 ± 0.2	0.973 ± 0.005	1.6 ± 0.4	**0.6 ± 0.2**	1.2 ± 0.3	100.0 ± 0.0
30 hours	reg. self-play (ours)	**0.0 ± 0.0**	0.3 ± 0.2	99.7 ± 0.2	**0.976 ± 0.005**	**1.4 ± 0.4**	0.8 ± 0.3	1.1 ± 0.3	99.9 ± 0.0

Self-play exposes agents to an evolving curriculum of partners. Regularized self-play agents train in a non-stationary environment: early in training, all partners exhibit near-random behavior; as training progresses, the partner distribution shifts toward increasingly competent policies. This implicit curriculum promotes robustness to diverse partner behavior. Accordingly, regularized self-play agents achieve low collision rates in both self-play and cross-play with human logs (below 1.5% in each setting). By contrast, SMART trained on 52 days of human data incurs a 6% self-play collision rate, far exceeding its 1.6% rate when paired with logs. This gap reflects two compounding factors: (1) the number of samples, as self-play RL policies train on 20 billion transitions versus 225 million for SMART (Figure 1); and (2) training paradigm, as SMART is first optimized open-loop for log-likelihood and then finetuned closed-loop to stay near the log distribution, and is never exposed to the diversity of partners that self-play RL naturally provides. An ablation isolates the contribution of the self-play training setting: agents trained *directly against the human-replay population* (single-

agent RL) perform well within that population (at-fault collision rate 0.2–0.3%) but degrade when evaluated in self-play (0.8–1.2%), consistent with the hypothesis that exposure to reactive and diverse partners during training contributes to robustness (analysis in Figure 19).

4.2 Behavior and Safety Analysis

The goal of this section is to understand the behavioral differences between vanilla and regularized self-play policies beyond straightforward performance metrics.

Regularized policies exhibit lower-severity collisions. Collision rates, as reported in Sections 4.1 and 4.3, measure how often agents fail, but not how bad the failures are. This distinction matters when policies are deployed alongside humans. Following Waymo's most recent safety report [27], we quantify collision severity via the *change in velocity at impact* (Δv), a widely studied proxy for occupant injury risk. As shown in Table 8 and Figure 3, regularization reduces both the frequency and the severity of failures. The mean per-event Δv drops from 2.09 m/s to 1.71 m/s, and the maximum observed impact velocity falls from 13.71 m/s to 8.09 m/s. The improvement is more apparent when we focus on the tail of collision events: 14.3% of unregularized collisions exceed 15 mph, the threshold above which serious injury risk rises substantially, compared to 7.5% for the regularized model. The survival curve in Figure 3 (right) shows the two groups are nearly indistinguishable at low Δv, with the gap opening sharply above 5 m/s and widening through the severe range. Regularization thus produces policies that not only collide less often but also cause less damage when they do collide.

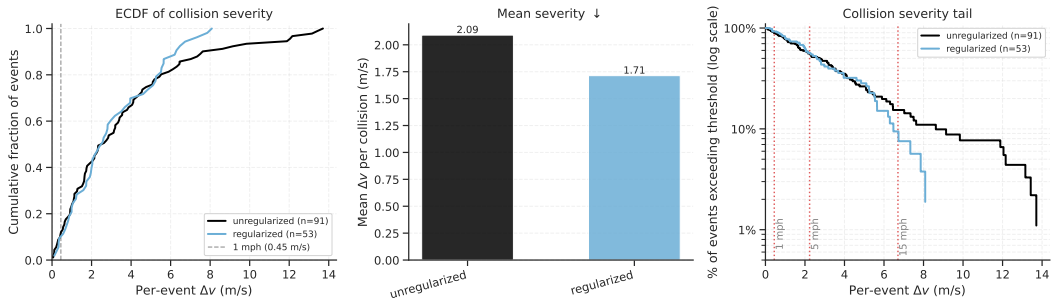

Figure 3: **Analyzing collision event severity.** *Left:* empirical CDF of per-event Δv. The dashed line marks Waymo's 1 mph (0.45 m/s) reporting threshold. *Center:* mean Δv per collision event, conditional on a collision occurring. Regularized collisions are on average 18% lower in severity (1.71 vs. 2.09 m/s). *Right:* fraction of collisions exceeding Δv (log scale).

Regularized self-play improves realism with minimal data. Vanilla self-play scores 0.680 on the WOSAC meta-score [26], with the largest deficits in the kinematic and interactive groups. Anchoring to 30 minutes of human data increases this to 0.725; the meta-score does not improve with additional data, suggesting BC anchor quality is the limiting factor. SMART-tiny CLSFT [11, 52] achieves the highest realism score (0.755), yet underperforms on collision rate and task completion across every data bin (Section 4.1), confirming that distributional similarity to human data does not necessarily imply safety or competence [60]. Full results are in Appendix E.5.

Regularized policies display more social driving behavior. We perform a qualitative analysis with representative videos available at https://sites.google.com/view/anonymous-human-like-autonomy/. The most salient difference is that regularized policies are more considerate of surrounding traffic: they maintain greater following distances, avoid cutting in, and yield at intersections relative to vanilla self-play agents. RL policies are trained to maximize the expected cumulative *discounted* return. An undesirable side-effect of this is that policies tend to achieve their task in the least number of steps possible. This is different than what humans do. A human driver will aim to get to her destination on time, but is not trying to get there as quickly as possible; *satisficing* [61] rather than *optimizing*. As visible in the videos and supported by the

average episode length, regularization partially corrects for this: regularized agents complete their episodes in 64 steps on average (± 3.5), compared to 38 (± 2.6) steps for vanilla self-play.

4.3 The Role of Scenario Metadata

Scenario diversity is essential for learning general policies. Beyond human driving data, a cheaper source of sim grounding data is scenario *metadata*: road graphs, initial positions, and velocities. The number of training scenarios (a proxy for map diversity) is a primary determinant of generalization, both to held-out maps and to the human-replay population. For instance, the at-fault collision rate of vanilla self-play drops from 46.2% with 10 scenarios to 4.1% with 10k. Regularized self-play follows the same trend at uniformly lower values. Typically, 10k-50k scenarios appear to be sufficient for near-perfect generalization from train to test maps. The full analysis is in Appendix E.2.

5 Conclusion & Limitations

Conclusion. We consider a series of experiments aimed at putting the mixing of human driving data with synthetic simulated experience on a more scientific footing. Our central finding is that a small amount of human data, roughly 30 minutes to 3 hours of human driving data, can dramatically move the needle towards human-compatible driving agents. This is three orders of magnitude less than SOTA imitation learning baselines and is achieved without reward engineering or domain randomization techniques. The broader implication is that when simulation is cheap, and some clear metrics for desirable behavior are available, human driving data may be best used **not** as the primary training signal but as a *lightweight anchor* that steers policies away from effective-but-alien equilibria.

This empirical result raises a deeper question that we have only touched the surface of: Given the availability of near-limitless simulated self-play experience, what is the *complementary value* of a bit of human data? Can we quantify this? We can loosely intuit that the resulting self-play RL policies are more robust since they are exposed to a broader coverage of the state space. For instance, the self-play agents learn from 20B transitions and start training from random play, whereas the IL baseline is trained on a fixed dataset of 225 million expert transitions (Figure 1). But count is a crude explanation; not all transitions are equally informative. Recent work on Epiplexity [62] takes a step toward formalizing this, conceptually defining the value of data as the amount of learnable structure it contains. However, translating such notions into practical tools for selecting data remains an open challenge.

Limitations and future work.

1. **Robustness in tight coordination scenarios**: We perform an additional analysis to better understand the limitations of the resulting regularized policies. We curate a small dataset consisting of the top 200 most difficult interactive scenarios (see Appendix D.2). Repeating the analysis from Section 4.1 on this set of harder scenarios shows that, while the ranking of the policies holds (reg. self-play RL policies still outperform the SMART and vanilla self-play baselines by the same margins), the absolute at-fault collision rate increases from 0.7% to 2.1-2.8%. This indicates that there is room for improvement in the robustness of the resulting policies. Arguably, not all of these contacts reflect policy failures: some are caused by replay agents cutting abruptly into the SDC's lane, leaving almost no physically feasible avoidance response. What constitutes a fair collision-avoidance benchmark beyond at-fault heuristics is itself a difficult open question in both industry and academia [63]. Nevertheless, an important direction for future work is to improve the robustness of regularized policies. See Appendix F for the results, an in-depth discussion, and ideas to improve along this axis.
2. **External validity of evals**: Our evaluations use human replays and IDM-controlled agents in simulation as proxies for coordination with humans. The extent to which performance in these settings transfers to on-road deployment remains an open question; closing this sim-to-real gap is left to future work.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[2] J. Perolat, B. De Vylder, D. Hennes, E. Tarassov, F. Strub, V. de Boer, P. Muller, J. T. Connor, N. Burch, T. Anthony, et al. Mastering the game of Stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

[3] S. Sokota, E. Vinitsky, H. Hu, J. Z. Kolter, and G. Farina. Superhuman ai for stratego using self-play reinforcement learning and test-time search. *arXiv preprint arXiv:2511.07312*, 2025.

[4] 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*, 2024.

[5] M. Cusumano-Towner, D. Hafner, A. Hertzberg, B. Huval, A. Petrenko, E. Vinitsky, E. Wijmans, T. Killian, S. Bowers, O. Sener, P. Krähenbühl, and V. Koltun. Robust autonomy emerges from self-play. *arXiv preprint arXiv:2502.03349*, 2025.

[6] D. Cornelisse, A. Pandya, K. Joseph, J. Suárez, and E. Vinitsky. Building reliable sim driving agents by scaling self-play. *arXiv preprint arXiv:2502.14706*, 2025.

[7] W.-J. Chang, A. Rangesh, K. Joseph, M. Strong, M. Tomizuka, Y. Hu, and W. Zhan. SPACeR: Self-play anchoring with centralized reference models. *arXiv preprint arXiv:2510.18060*, 2025.

[8] T. Yin, Z. Mei, Z. Zheng, M. Yamane, D. Wang, J. Sceats, S. M. Bateman, L. Zha, A. Badithela, O. Shorinwa, and A. Majumdar. PlayWorld: Learning robot world models from autonomous play. *arXiv preprint arXiv:2603.09030*, 2026.

[9] J. Z. Leibo, E. Hughes, M. Lanctot, and T. Graepel. Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research. *arXiv preprint arXiv:1903.00742*, 2019.

[10] A. Bakhtin, D. Wu, A. Lerer, and N. Brown. No-press diplomacy from scratch. *Advances in Neural Information Processing Systems*, 34:18063–18074, 2021.

[11] W. Wu, X. Feng, Z. Gao, and Y. Kan. Smart: Scalable multi-agent real-time motion generation via next-token prediction. *Advances in Neural Information Processing Systems*, 37:114048–114071, 2024.

[12] J. Qiu, A. Saviolo, C. Wang, M. Wang, and X. Huang. Heterogeneous self-play for realistic highway traffic simulation. 2026. URL https://arxiv.org/abs/2604.16406.

[13] W. B. Knox, A. Allievi, H. Banzhaf, F. Schmitt, and P. Stone. Reward (mis) design for autonomous driving. *Artificial Intelligence*, 316:103829, 2023.

[14] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[15] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[16] J. Philion, X. B. Peng, and S. Fidler. Trajeglish: Traffic modeling as next-token prediction. *arXiv preprint arXiv:2312.04535*, 2023.

[17] M. Baniodeh, K. Goel, S. Ettinger, C. Fuertes, A. Seff, T. Shen, C. Gulino, C. Yang, G. Jerfel, D. Choe, R. Wang, V. Kallem, S. Casas, R. Al-Rfou, B. Sapp, and D. Anguelov. Scaling laws of motion forecasting and planning: A technical report. *arXiv preprint arXiv:2506.08228*, 2025.

[18] J. Suarez. PufferLib: Making reinforcement learning libraries and environments play nice. *arXiv preprint arXiv:2406.12905*, 2024.

[19] D. Cornelisse, S. Cheng, P. Mandavilli, J. Hunt, K. Joseph, W. Doulazmi, V. Charraut, A. Gupta, J. Suarez, and E. Vinitsky. PufferDrive: A fast and friendly driving simulator for training and evaluating RL agents, 2025. URL https://github.com/Emerge-Lab/PufferDrive.

[20] H. Hu, D. J. Wu, A. Lerer, J. Foerster, and N. Brown. Human-ai coordination via human-regularized search and learning. *arXiv preprint arXiv:2210.05125*, 2022.

[21] A. Bakhtin, D. J. Wu, A. Lerer, J. Gray, A. P. Jacob, G. Farina, A. H. Miller, and N. Brown. Mastering the game of no-press Diplomacy via human-regularized reinforcement learning and planning. In *International Conference on Learning Representations*, 2023. arXiv:2210.05492.

[22] D. Cornelisse and E. Vinitsky. Human-compatible driving partners through data-regularized self-play reinforcement learning. In *Reinforcement Learning Journal*, 2024. arXiv:2403.19648.

[23] Z. Wang, S. Rahmani, D. Cornelisse, B. Sarkar, A. D. Goldie, J. N. Foerster, and S. Whiteson. Learning to drive in new cities without human demonstrations. *arXiv preprint arXiv:2602.15891*, 2026.

[24] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9710–9719, 2021.

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

[26] N. Montali, J. Lambert, P. Mougin, A. Kuefler, N. Rhinehart, M. Li, C. Gulino, T. Emrich, Z. Yang, S. Whiteson, et al. The waymo open sim agents challenge. *Advances in Neural Information Processing Systems*, 36:59151–59171, 2023.

[27] Waymo LLC. Waymo safety impact. https://waymo.com/safety/impact/, 2025. Accessed: 2026-05-06.

[28] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li. End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46 (12):10164–10183, 2024.

[29] X. Jia, Z. Yang, Q. Li, Z. Zhang, and J. Yan. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. *Advances in Neural Information Processing Systems*, 37:819–844, 2024.

[30] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17853–17862, 2023.

[31] B. Jiang, S. Chen, Q. Xu, B. Liao, J. Chen, H. Zhou, Q. Zhang, W. Liu, C. Huang, and X. Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8350, 2023.

[32] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, and H. Chen. A survey on trajectory-prediction methods for autonomous driving. *IEEE transactions on intelligent vehicles*, 7(3):652–674, 2022.

[33] E. Vinitsky, N. Lichtlé, S. Kanaa, et al. Nocturne: a scalable driving benchmark for bringing multi-agent learning one step closer to the real world. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[34] H. Caesar, V. Bankiti, A. Lang, S. Vora, V. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. arxiv. 2019.

[35] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023.

[36] M. Bansal, A. Krizhevsky, and A. Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018.

[37] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer, 2020.

[38] J. Gu, C. Sun, and H. Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021.

[39] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp. Wayformer: Motion forecasting via simple & efficient attention networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2980–2987. IEEE, 2023.

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

[41] Z. Zhou, L. Ye, J. Wang, K. Wu, and K. Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8823–8833, 2022.

[42] S. Shi, L. Jiang, D. Dai, and B. Schiele. Motion transformer with global intention localization and local movement refinement. *Advances in Neural Information Processing Systems*, 35: 6531–6543, 2022.

[43] Z. Zhou, J. Wang, Y.-H. Li, and Y.-K. Huang. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17863–17873, 2023.

[44] A. Seff, B. Cera, D. Chen, M. Ng, A. Zhou, N. Nayakanti, K. S. Refaat, R. Al-Rfou, and B. Sapp. Motionlm: Multi-agent motion forecasting as language modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8579–8590, 2023.

[45] Z. Zhong, D. Rempe, D. Xu, Y. Chen, S. Veer, T. Che, B. Ray, and M. Pavone. Guided conditional diffusion for controllable traffic simulation. In *2023 IEEE international conference on robotics and automation (ICRA)*, pages 3560–3566. IEEE, 2023.

[46] C. M. Jiang, Y. Bai, A. Cornman, C. Davis, X. Huang, H. Jeon, S. Kulshrestha, J. Lambert, S. Li, X. Zhou, et al. Scenediffuser: Efficient and controllable driving simulation initialization and rollout. *Advances in Neural Information Processing Systems*, 37:55729–55760, 2024.

[47] Z. Huang, Z. Zhang, A. Vaidya, Y. Chen, C. Lv, and J. F. Fisac. Versatile behavior diffusion for generalized traffic agent simulation. *arXiv preprint arXiv:2404.02524*, 2024.

[48] S. Tan, J. Lambert, H. Jeon, S. Kulshrestha, Y. Bai, J. Luo, D. Anguelov, M. Tan, and C. M. Jiang. Scenediffuser++: City-scale traffic simulation via a generative world model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1570–1580, 2025.

[49] B. Liao, S. Chen, H. Yin, B. Jiang, C. Wang, S. Yan, X. Zhang, X. Li, Y. Zhang, Q. Zhang, et al. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12037–12047, 2025.

[50] Y. Lu, J. Fu, G. Tucker, X. Pan, E. Bronstein, R. Roelofs, B. Sapp, B. White, A. Faust, S. Whiteson, et al. Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7553–7560. IEEE, 2023.

[51] Z. Peng, W. Luo, Y. Lu, T. Shen, C. Gulino, A. Seff, and J. Fu. Improving agent behaviors with RL fine-tuning for autonomous driving. In *Computer Vision - ECCV 2024 - 18th European Conference*, volume 15083 of *Lecture Notes in Computer Science*, pages 165–181. Springer, 2024.

[52] Z. Zhang, P. Karkus, M. Igl, W. Ding, Y. Chen, B. Ivanovic, and M. Pavone. Closed-loop supervised fine-tuning of tokenized traffic models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[53] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[54] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

[55] H. Hu, A. Lerer, A. Peysakhovich, and J. Foerster. "other-play" for zero-shot coordination. In *International conference on machine learning*, pages 4399–4410. PMLR, 2020.

[56] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020.

[57] A. P. Jacob, D. J. Wu, G. Farina, A. Lerer, H. Hu, A. Bakhtin, J. Andreas, and N. Brown. Modeling strong and human-like gameplay with KL-regularized search. In *International Conference on Machine Learning*, pages 9695–9728. PMLR, 2022.

[58] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, Aug. 2000. ISSN 1063-651X, 1095-3787. doi:10.1103/PhysRevE.62.1805. URL https://link.aps.org/doi/10.1103/PhysRevE.62.1805.

[59] V. Charraut, W. Doulazmi, T. Tournaire, and T. Buhet. V-Max: A RL framework for autonomous driving. *Reinforcement Learning Journal*, 6:2427–2451, 2025.

[60] D. Cornelisse. Human-likeness metrics for autonomous agents: are we measuring the right thing? Substack, 2025. Blog post analyzing the Waymo Open Sim Agent Challenge (WOSAC) realism benchmark.

[61] D. Arumugam, S. Kumar, R. Gummadi, and B. Van Roy. Satisficing exploration for deep reinforcement learning. *arXiv preprint arXiv:2407.12185*, 2024.

[62] M. Finzi, S. Qiu, Y. Jiang, P. Izmailov, J. Z. Kolter, and A. G. Wilson. From entropy to epiplexity: Rethinking information for computationally bounded intelligence. *arXiv preprint arXiv:2601.03220*, 2026.

[63] J. M. Scanlon, K. D. Kusano, J. Engstrom, and T. Victor. Collision avoidance effectiveness of an automated driving system using a human driver behavior reference model in reconstructed fatal collisions. In *WCX SAE World Congress Experience*. SAE Technical Paper, 2026.

[64] D. Dauner, M. Hallgarten, T. Li, X. Weng, Z. Huang, Z. Yang, H. Li, I. Gilitschenski, B. Ivanovic, M. Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *Advances in Neural Information Processing Systems*, 37:28706–28719, 2024.

[65] A. Distelzweig, F. Janjoš, A. Look, A. Rothenhäusler, D. Jost, O. Scheel, R. Rajan, D. Cornelisse, E. Vinitsky, and J. Boedecker. Beyond self-play and scale: A behavior benchmark for generalization in autonomous driving. *arXiv preprint arXiv:2605.10034*, 2026.

[66] C. Gulino, J. Fu, W. Luo, G. Tucker, E. Bronstein, Y. Lu, J. Harb, X. Pan, Y. Wang, X. Chen, et al. Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. *Advances in Neural Information Processing Systems*, 36:7730–7742, 2023.

[67] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*, 2021.

A Simulation Environment and Design

A.1 World Initialization from Scenario Metadata

We use PufferDrive 2.0 for simulation and training [19]. PufferDrive is a batched simulator that runs many environments in parallel, reaching 390k steps per second (SPS) on an NVIDIA RTX 5090 GPU. We initialize environments using the Waymo Open Motion Dataset (WOMD) [24], which provides a large set of multi-agent traffic scenarios. Each scenario supplies the metadata we need: the roadgraph, a variable number of agents (cars, cyclists, and pedestrians), and other objects in the scene. This information is the output of a perception stack, so we operate directly on these clean features (in bounding-box world).

Each scenario is 9 seconds long and discretized into 90 steps. We take each logged agent's initial position ($t = 0$) as its starting position in the scene, and its last valid logged position ($t = T$) as its goal, which lets us goal-condition the agents. The full Waymo training dataset contains 500k scenarios, but in this paper we use at most 50k of the randomly sampled scenarios. When constructing the environments, we randomly sample scenarios from WOMD until we hit a target number of agents (e.g., on an NVIDIA RTX 4080 with 16GB of memory, we keep adding environments until we reach 1024 agents).

A.2 Observation Space

We take a decentralized approach and provide every agent with a partial view of the environment in a local coordinate frame. This is similar to the observation space of prior related works, such as GIGAFLOW [5], and GPUDrive [4]. At each timestep, an agent receives the combination of three feature blocks: an ego block describing its own state, a partner block describing the $N_p = 31$ closest other agents within a 50 m radius, and a road block describing up to $N_r = 128$ nearby road segments drawn from a 21×21 grid of $5\,\text{m} \times 5\,\text{m}$ cells centered on the agent. Missing slots (fewer partners or road segments than the maximum) are zero-padded. Tables 2, 3, and 4 list the features in each block. All positions and headings are expressed in the agent's local frame, so the observation is invariant to the global pose of the scene. The total observation vector has dimension $11 + 7 \times 31 + 7 \times 128 = 1{,}124$.

Table 2: Ego features (14 values) for the delta-local dynamics model. Features 0–3 expose the sampled conditioning variables to the policy so it can modulate its behavior as a function of λ and the reward weights (Section 4.3). This was not used further in the paper; all values are kept at a fixed value.

Idx	Feature	Normalization	Description
0	λ	—	Human-regularization coefficient
1	r_{coll}	—	Sampled collision reward
2	r_{off}	—	Sampled off-road reward
3	r_{goal}	—	Sampled goal reward
4	Δx_{goal}	$\times 0.005$	Goal position (ego frame), longitudinal
5	Δy_{goal}	$\times 0.005$	Goal position (ego frame), lateral
6	signed speed	$/\,100$ m/s	Speed projected onto heading
7	vehicle width	$/\,15$ m	Ego bounding-box width
8	vehicle length	$/\,30$ m	Ego bounding-box length
9	collision flag	$\{0, 1\}$	1 if currently colliding
10	entity type	$/\,3$	Vehicle (1), pedestrian (2), cyclist (3)

Table 3: Partner features (7 values $\times$ 31 partners = 217 values). Partners are ordered by index and filtered to those within 50 m of the ego agent. All positions and headings are in the ego frame.

Idx	Feature	Normalization	Description
0	Δx	$\times 0.02$	Partner position, longitudinal
1	Δy	$\times 0.02$	Partner position, lateral
2	partner width	/ 15 m	Partner bounding-box width
3	partner length	/ 30 m	Partner bounding-box length
4	$\cos(\Delta\psi)$	—	Relative heading, cosine component
5	$\sin(\Delta\psi)$	—	Relative heading, sine component
6	partner signed speed	/ 100 m/s	Signed speed along partner's heading

Table 4: Road-segment features (7 values $\times$ 128 segments = 896 values). Segments are drawn from a 21×21 grid of 5 m cells centered on the ego agent, and include road lanes, road lines, and road edges. Each segment is described by the midpoint, length, and orientation of a single polyline segment.

Idx	Feature	Normalization	Description
0	midpoint x	$\times 0.02$	Segment midpoint, longitudinal (ego frame)
1	midpoint y	$\times 0.02$	Segment midpoint, lateral (ego frame)
2	segment length	/ 100 m	Length of the polyline segment
3	segment width	/ 100 m	Fixed nominal width (0.1 m)
4	$\cos(\theta)$	—	Segment orientation in ego frame
5	$\sin(\theta)$	—	Segment orientation in ego frame
6	segment type	$\{0, 1, 2\}$	Road lane (0), road line (1), road edge (2)

A.3 Actions and Dynamics

We use a single dynamics model with a discretized action space for both the unregularized and regularized agents.

Delta-local dynamics with kinematic constraints. The action is a triple $(\Delta x, \Delta y, \Delta\psi)$ in the agent's local frame at time t. Translation is rotated into the world frame and added to the position; heading is updated directly:

$$x_{t+1} = x_t + \cos(\psi_t)\,\Delta x - \sin(\psi_t)\,\Delta y, \tag{3}$$

$$y_{t+1} = y_t + \sin(\psi_t)\,\Delta x + \cos(\psi_t)\,\Delta y, \tag{4}$$

$$\psi_{t+1} = \mathrm{wrap}(\psi_t + \Delta\psi). \tag{5}$$

Velocity is reported as the world-frame displacement divided by $\Delta t = 0.1$ s. We bound each component roughly based on realistic actions present in the human data, as shown in Figure 4; specifically, we define $\Delta x \in [-3.5, 3.5]$ m, $\Delta y \in [-0.1, 0.1]$ m, and $\Delta\psi \in [-\pi/6, \pi/6]$. Each of the three dimensions is binned independently into 51, 51, and 127 values, respectively. Figure 4 shows that the distributions for Δy and $\Delta\psi$ are roughly symmetric, whereas the distribution for Δx is strongly asymmetric. This is expected, since most vehicles move forward and only a small number of agents in the scenes drive in reverse (e.g., when parking).

Delta-local dynamics are kinematically unconstrained by default: the agent can translate laterally without rotating, pivot in place, or instantaneously reverse its heading rate. To prevent impossible behaviors, we apply two physics-based constraints to the action at each step. Each constraint clips the action after the previous one has been applied, with the previously executed (post-constraint) values used as the reference. The constraints are:

1. **Longitudinal acceleration bound.** The change in implied forward speed is clipped to $\pm A_{\text{long,max}} \cdot \Delta t$, where $A_{\text{long,max}} = 8\,\text{m/s}^2$. This caps acceleration and braking.

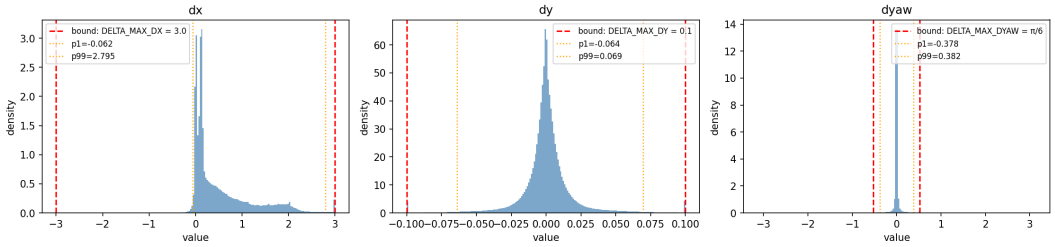

Figure 4: Discretized delta-local action space for each component (Δx, Δy, $\Delta\psi$). Histograms show the empirical density (blue) of 10,996,751 valid action timesteps recovered from expert trajectories across 10,000 maps. Yellow lines mark the 1st and 99th percentiles of the data; red lines mark the action-space bounds (± 3.5 m, ± 0.1 m, $\pm\pi/6$ rad). Each dimension is binned independently into 512 values. The bounds were chosen to respect natural movements in the data: 0.00% of Δx and Δy samples fall outside their bounds, and 0.71% of $\Delta\psi$ samples fall outside $\pm\pi/6$.

2. **Lateral motion envelope.** Lateral displacement is bounded by $|\Delta y| \leq |\Delta x| \cdot \tan(\delta_{\max})$, where $\delta_{\max} = 0.7$ rad is the maximum effective steering angle. This eliminates lateral sliding and side-shimmy at low forward speed.

These physical constraints prevent kinematically implausible actions; they do not encode any preference over driving style and are independent of the human anchor.

A.4 Collecting Human Driving Data

The behavioral cloning (BC) anchor is trained on observation–action pairs (o_t, a_t). We therefore need actions that (i) live in the simulator's action space and (ii) reproduce the logged motion when applied through the simulator's dynamics. We construct the dataset in two steps. Figure 6 shows three examples of this process in the simulator.

Step 1: Inferring actions from the data. For each timestep t, we invert the delta-local dynamics to recover the action that produced the next logged state. Projecting the world-frame displacement into the agent's local frame at t gives:

$$\Delta x_t = \cos(\psi_t)(x_{t+1} - x_t) + \sin(\psi_t)(y_{t+1} - y_t), \tag{6}$$

$$\Delta y_t = -\sin(\psi_t)(x_{t+1} - x_t) + \cos(\psi_t)(y_{t+1} - y_t), \tag{7}$$

$$\Delta\psi_t = \mathrm{wrap}(\psi_{t+1} - \psi_t). \tag{8}$$

Each triple is clipped to the action bounds and snapped to the nearest discrete bin. Timesteps where either t or $t+1$ is flagged invalid in the log are marked as missing and excluded from training.

Step 2: Replaying actions through the simulator. To produce observations, we replay the inferred action sequence through the simulator and record the observation at every resulting state. The BC anchor is then trained on the resulting (simulator observation, inferred action) pairs. Discretization introduces a small error that grows inversely with bin size (details below); to prevent its accumulation, we instead *teleport* agents to each successive state rather than stepping them forward with the inferred actions. We note that stepping agents directly is also viable when using larger action spaces, where the discretization error is smaller.

Effect of discretization on performance. Figure 5 and Table 5 quantify the cost of discretization. Continuous actions reproduce the logged trajectory almost exactly (ADE 0.001 m), confirming that the delta-local dynamics and kinematic constraints are themselves well-posed. Discretizing into 512 bins per dimension introduces a quantization floor of ADE 0.097 m, which is roughly two orders of magnitude larger, but is still very close to the original trajectory. Off-road and collision rates increase

modestly under discretization (1.2% vs. 0.8% off-road, 0.4% vs. 0.0% collision), reflecting the rare cases where snapping to the nearest bin pushes the SDC just outside a road edge or into a static neighbor; both representations complete the route in 100% of scenarios.

Table 5: Inferred-expert-action quality for the delta-local dynamics model. Comparison of discrete (bin-quantized) vs continuous (direct float) expert actions. Aggregated over 10,240 pooled samples. Values are mean ± SE.

Action type	Route prog. (%) ↑	Coll. (%) ↓	Off-road (%) ↓	ADE (m) ↓	Lateral L2 (m) ↓	Longitudinal L2 (m) ↓
discrete	100.0	0.4 ± 0.1	1.2 ± 0.2	0.097 ± 0.002	0.096 ± 0.002	0.004 ± 0.000
continuous	100.0	0.0	0.8 ± 0.1	0.001 ± 0.000	0.001 ± 0.000	0.001 ± 0.000

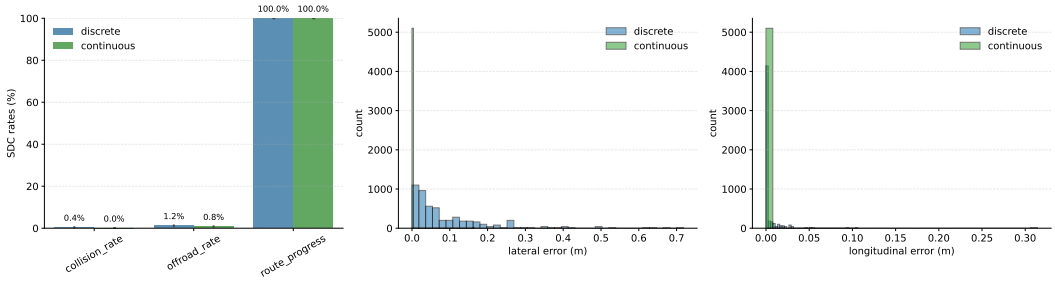

Figure 5: Effect of action discretization on inferred-expert-action quality. We replay each agent's logged trajectory through the simulator using actions inferred from the logs, comparing discrete (bin-quantized, blue) and continuous (direct float, green) action representations. **Left:** SDC rates aggregated across 10,240 pooled samples; both representations complete the route in 100% of scenarios, but discretization induces modestly higher off-road and collision rates. **Center, right:** distributions of per-trajectory lateral and longitudinal L2 error to the logged pose. Continuous actions reproduce the log almost exactly (errors concentrated near zero), while discrete actions exhibit a small but consistent quantization floor of ~ 0.1 m laterally. Error bars on the bar plot denote standard error.

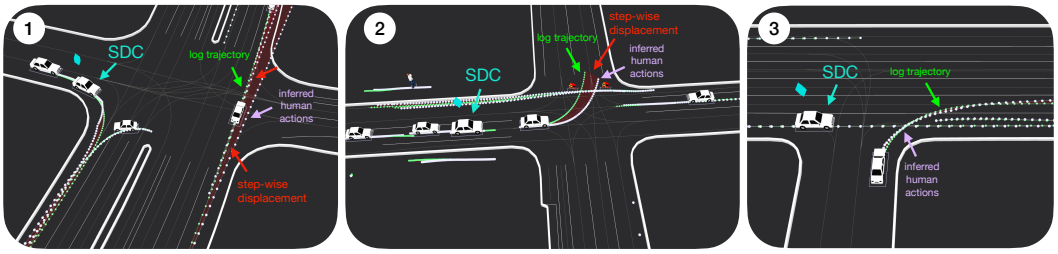

Figure 6: Three annotated example scenarios illustrating the human data collection process. The self-driving car (SDC), marked in cyan, is the Waymo vehicle whose human-driven trajectory we use as the driving log. Logged trajectories are shown in green; purple trajectories show the result of stepping each agent through the simulator under the inferred delta-local actions. We select only the SDC trajectory because it is typically the cleanest data in the scene; the visualized step-wise displacement illustrates a few low-quality (high-ADE) log trajectories that would otherwise contaminate the anchor.

Table 6: BC anchor evaluation. Open-loop metrics on the held-out validation set; closed-loop metrics averaged over validation scenes. Within-5-bin accuracy is the average of Δx, Δy, Δyaw accuracies at the final training step.

	Open-loop			Closed-loop self-play		Closed-loop human-replay (SDC only)	
Human data (h)	Acc. (%)	Acc. ± 5 bins (%)	Loss	Route prog.	Score	Route prog.	Score
0.2	23.4	72.4	15.677	0.720 ± 0.012	0.215 ± 0.013	0.765 ± 0.007	0.242 ± 0.009
0.5	36.1	87.3	5.269	0.719 ± 0.011	0.277 ± 0.014	0.800 ± 0.006	0.371 ± 0.011
3.0	48.2	92.6	1.641	0.835 ± 0.010	0.502 ± 0.017	0.842 ± 0.006	0.538 ± 0.011
30.0	**52.8**	**94.9**	**1.266**	$\mathbf{0.932 \pm 0.007}$	$\mathbf{0.685 \pm 0.016}$	$\mathbf{0.873 \pm 0.006}$	$\mathbf{0.666 \pm 0.010}$

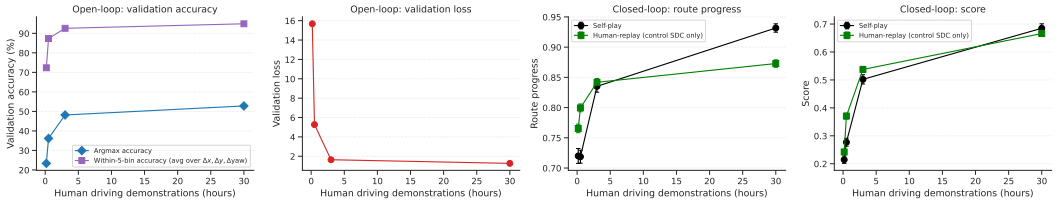

Figure 7: Open- and closed-loop performance of the anchor BC policies as a function of human driving data. Left: The final real (blue) and within 5 bin accuracy (purple) accuracy on 10,000 held-out validation scenarios. Right: Final validation loss. Right; Route progress; Right Score.

A.5 Reward Function

We use a sparse reward: $r^i = +1$ if agent i reaches its goal within $\delta = 2$ meters before the episode ends, -1 on collision or going off-road, and 0 otherwise. We deliberately omit dense shaping terms so that safe and human-compatible behaviors can emerge from regularization.

B Training

B.1 Behavioral Cloning Anchor Policies

Each anchor τ_n is trained by minimizing the negative log-likelihood of the logged actions under the factorized discrete action distribution described in Appendix A.3. We extract observation, action tuples through the procedure described in Appendix A.4. Note that we use only the SDC trajectory from each scenario for training, as it is the highest-quality data source. Since other agents are reconstructed from the perception stack, they exhibit more noise. Moreover, we have no guarantees about the driving quality of the surrounding humans. Since we obtain one trajectory per scene, each scenario contributes roughly 9 seconds of human data. Although these trajectories were collected in Waymo vehicles, they reflect manual human driving by an expert driver behind the wheel [24].

We train with Adam at a learning rate of 10^{-4} and a batch size of 2048 for up to 5000 epochs, with early stopping on the held-out validation loss after 100 epochs without improvement. Table 6 reports open- and closed-loop metrics for each anchor on 10,000 held-out validation scenarios. Figure 8 shows the 5-bin validation accuracy for each action head over training; from only 30 minutes of data, validation accuracy converges to between 80% and 90%. This metric is informative since there are 256 bins per action head, so the step sizes are very small.

Figures 10 and 9 compare the learned action distributions against the empirical distribution of the logged actions, for anchors trained on 30 minutes and 30 hours of data, respectively; in both cases, the learned distributions match the data reasonably well.

B.2 Self-Play Reinforcement Learning

Both self-play variants run for 20 billion steps.

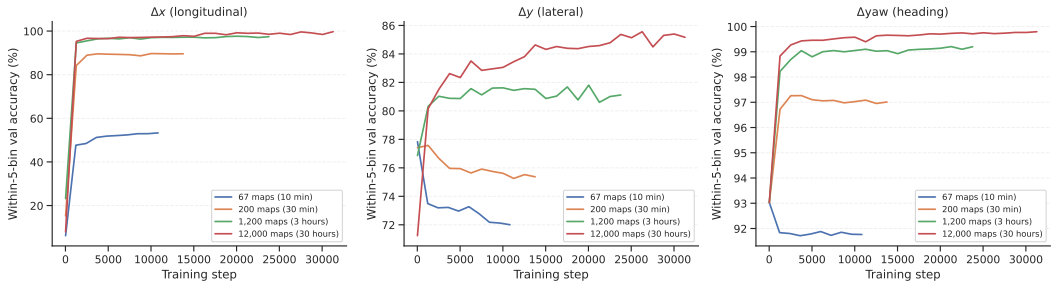

Figure 8: Training curves for the anchor policies. Each panel shows within-5-bin validation accuracy on a held-out set of scenarios for one action component (Δx, Δy, $\Delta\psi$). Curves terminate at different step counts because training stops once validation accuracy plateaus (no improvement for 100 consecutive epochs).

B.2.1 Regularization

Let π_θ denote the RL policy and τ_n the fixed BC anchor trained on n scenarios. We regularize π_θ toward τ_n by adding a KL penalty on states visited during the rollout:

$$\mathcal{L}_{\text{reg}}(\theta) = \frac{\lambda}{M} \sum_{j=1}^{M} D_{\text{KL}}(\tau_n(\cdot \mid o_j) \,\|\, \pi_\theta(\cdot \mid o_j)) \,, \tag{9}$$

where $\lambda = 0.075$ is fixed throughout training and inference and M is the minibatch size. The full objective augments standard PPO with this penalty:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{pg}} + c_v \, \mathcal{L}_{\text{v}} - c_H \, H + \mathcal{L}_{\text{reg}}, \tag{10}$$

where $\mathcal{L}_{\text{pg}}$ is the clipped surrogate policy-gradient loss, $\mathcal{L}_{\text{v}}$ the value-function loss, H the entropy bonus, and c_v, c_H their respective coefficients. The KL term pulls π_θ toward the anchor on states the policy actually visits, rather than on the offline logged data distribution. Setting $\lambda = 0$ recovers unregularized self-play.

B.2.2 Hyperparameters

Table 7 lists the hyperparameters. We use the same parameters for regularized self-play RL and the baseline.

Table 7: PPO training hyperparameters.

Architecture		*Training*		*Environment & Rewards*	
Input size	64	Total timesteps	20B	Number of agents	1,024
Hidden size	256	Batch size	524,288	Number of workers	16
RNN type	LSTM	Minibatch size	32,768	Episode length	150 steps
RNN input size	256	Rollout horizon	32	Timestep Δt	0.1 s
RNN hidden size	256	Update epochs	1	Goal radius	2.0 m
		Learning rate	4.26×10^{-3}	Action space	Discrete
		LR schedule	Linear annealing	Dynamics model	Delta-local
		Adam β_1	0.9	Goal reward	$+1.0$
		Adam β_2	0.999	Collision penalty	-1.0
		Adam ϵ	10^{-8}	Off-road penalty	-1.0
		Clip coefficient	0.2		
		Entropy coefficient	0.001		
		VF coefficient	2.0		
		VF clip	0.2		
		GAE λ	0.95		
		Discount γ	0.99		
		Max gradient norm	1.0		
		Priority α	0.85		
		Priority β_0	0.85		
		V-trace c clip	1.0		
		V-trace ρ clip	1.0		
		Optimizer	Adam		
		Seed	42		

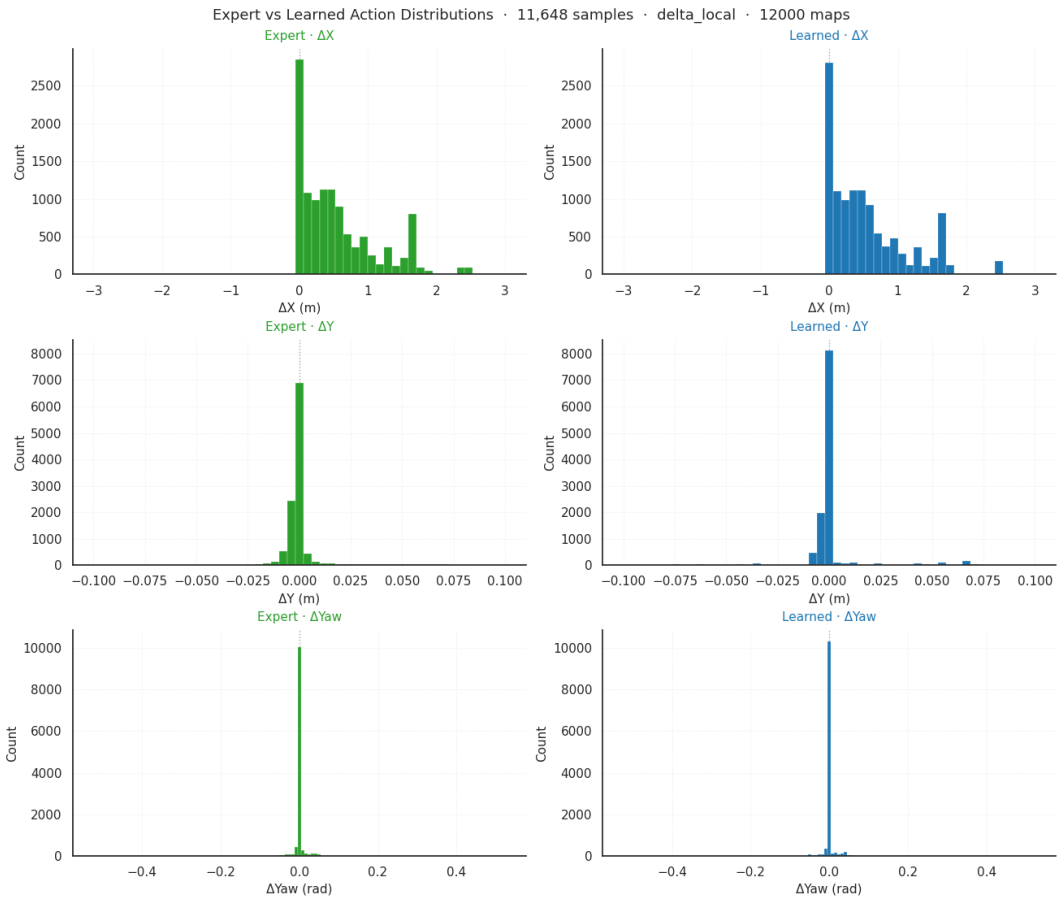

Figure 9: Example of actual vs. learned distributions - for 12k maps (30 hours)

B.3 SMART Model Training

We trained SMART models in CATK on subsets of the Waymo Open Motion Dataset (WOMD). WOMD motion shards were preprocessed into CATK's per-scenario cached format, and all training subsets were built from these cached scenario files. Our final local runs used `smart_mini_3M` with vehicle-only supervision on deterministic subsets of 67, 200, 1200, and 12000 scenarios. In the subset construction scripts we used, scenarios are sorted by cached scenario filename in lexicographic order. Vehicle-only supervision means that only vehicle agents contributed to the training loss, while pedestrians and cyclists were still present in the scene and available as context to the model. The local models were trained with CATK's `pre_bc` configuration on a single GPU for 64 epochs with batch size 8. These runs used standard supervised behavioral cloning and were not further fine-tuned with CAT-K / CLSFT. We additionally compare against two author-provided checkpoints: a behavioral-cloning checkpoint (`pre_bc_E31.ckpt`) and a closed-loop supervised fine-tuned checkpoint (`clsft_E9.ckpt`). For downstream evaluation, we exported predictions as `.pkl` files on the 10k validation scenes from the `pufferdrive_womd_subsets` dataset. We used two export modes: an all-agents mode and a planning mode in which only the SDC is controlled by the model and all other agents are replayed from ground truth. We re-exported all model and export-mode combinations with 32 rollouts for multimodal evaluation. We verified that the scenario-ID overlap between each local training subset (67, 200, 1200, and 12000) and the evaluation set is zero.

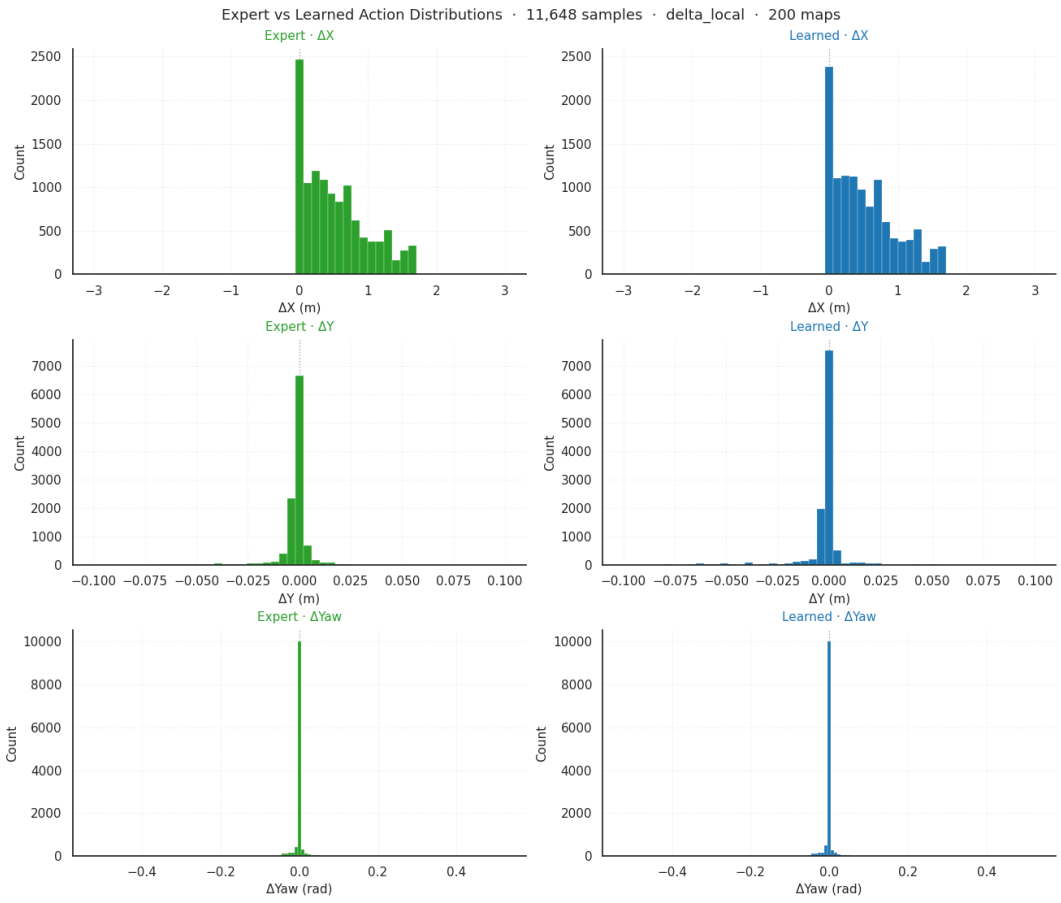

Figure 10: Example of actual vs. learned distributions - for 200 maps (30 min)

C Neural Network Architecture

Both the BC anchor and the RL policy share the same multi-modal encoder structure. The flattened environment observation vector is first unpacked into its modalities: ego state, partner agents, and road segments. Each modality is processed by a two-layer MLP with ReLU activation and layer normalisation between the two linear layers. Partner and road embeddings are then aggregated across objects via max-pooling, producing one vector per stream. The three pooled vectors are concatenated and passed through a shared two-layer MLP (Linear $\rightarrow$ ReLU $\rightarrow$ Linear) to produce the final embedding. Separate linear heads decode this embedding into logits over each action dimension; a separate linear head with unit output produces the value estimate. The two architectures differ in width and in the presence of recurrence:

- **BC anchor.** Per-stream MLP width 128, shared MLP $3\times128 \rightarrow 512 \rightarrow 512$. No recurrence. Actor heads are linear projections from the 512-dimensional embedding. It has 776,190 trainable parameters.
- **RL policy.** Per-stream MLP width 64, shared MLP $3\times64 \rightarrow 256 \rightarrow 256$. The 256-dimensional embedding is passed through a single-layer LSTM with input size 256 and hidden size 256 (PufferLib `LSTMWrapper`). Actor and critic heads are linear projections from the 256-dimensional LSTM output. It has 650k trainable parameters.

Road segment features include a categorical type field that is replaced by a 7-class one-hot vector before encoding, expanding the road feature dimension from d_{road} to $d_{\text{road}} + 6$.

D Evaluation

D.1 Settings

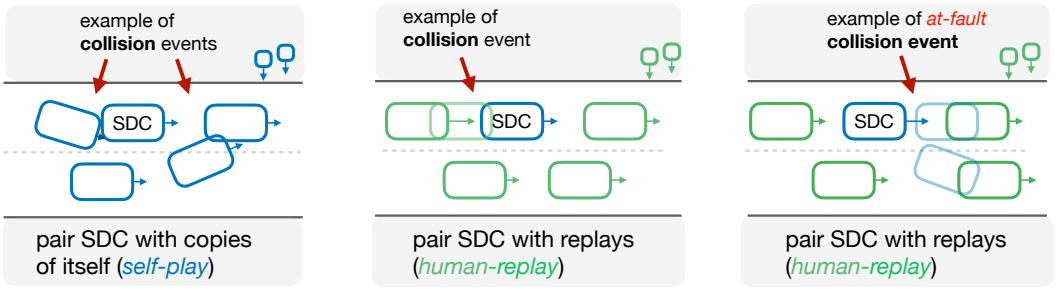

Figure 11: **Evaluation settings.** Self-play (left) and human-replay (center, right). Red arrows mark collisions. Rectangles are vehicles; squares are pedestrians. In human-replay, some collisions are effectively unavoidable: replay agents follow their logged trajectories and can drive into the controlled SDC from behind. We therefore distinguish between *collisions* (any contact) and *at-fault collisions* (contact caused by the controlled agent, following the NAVSIM benchmark [64]).

D.2 Filtering the Waymo Dataset for Interactive SDC Scenarios

As pointed out in earlier works [65, 22], many scenarios in the Waymo Open Motion Dataset (WOMD) involve the self-driving car (SDC) traveling without meaningful interaction with other agents—the SDC reaches its destination without requiring coordination or yielding. To increase the signal in our human-replay evaluation, we filter the dataset for scenarios in which the SDC trajectory intersects with other agents' trajectories, indicating situations that require coordination, such as merging, yielding, or navigating busy intersections.

We score each scenario by counting the number of segment-level intersections between the SDC trajectory and all other agent trajectories, optionally filtering crossings that meet a minimum acute-angle threshold (to exclude near-parallel overlaps, such as lane changes). From a pool of 10,000 held-out validation scenarios, we rank by intersection count and select the top 200 most interactive scenes. Figure 12 shows the resulting intersection count distributions across the full dataset and the selected subset, and Figure 13 shows nine representative examples from the selected set.

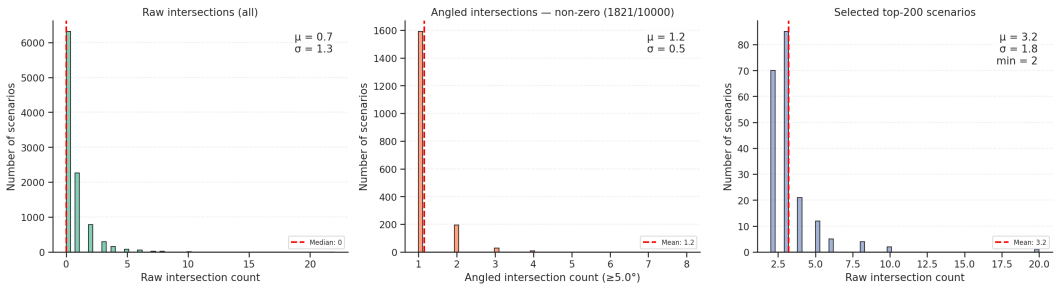

Figure 12: Distribution of SDC trajectory intersection counts. **Left:** raw intersection counts across all 50k scenarios. **Center:** angled intersections (non-zero only). **Right:** distribution within the selected top-200 subset.

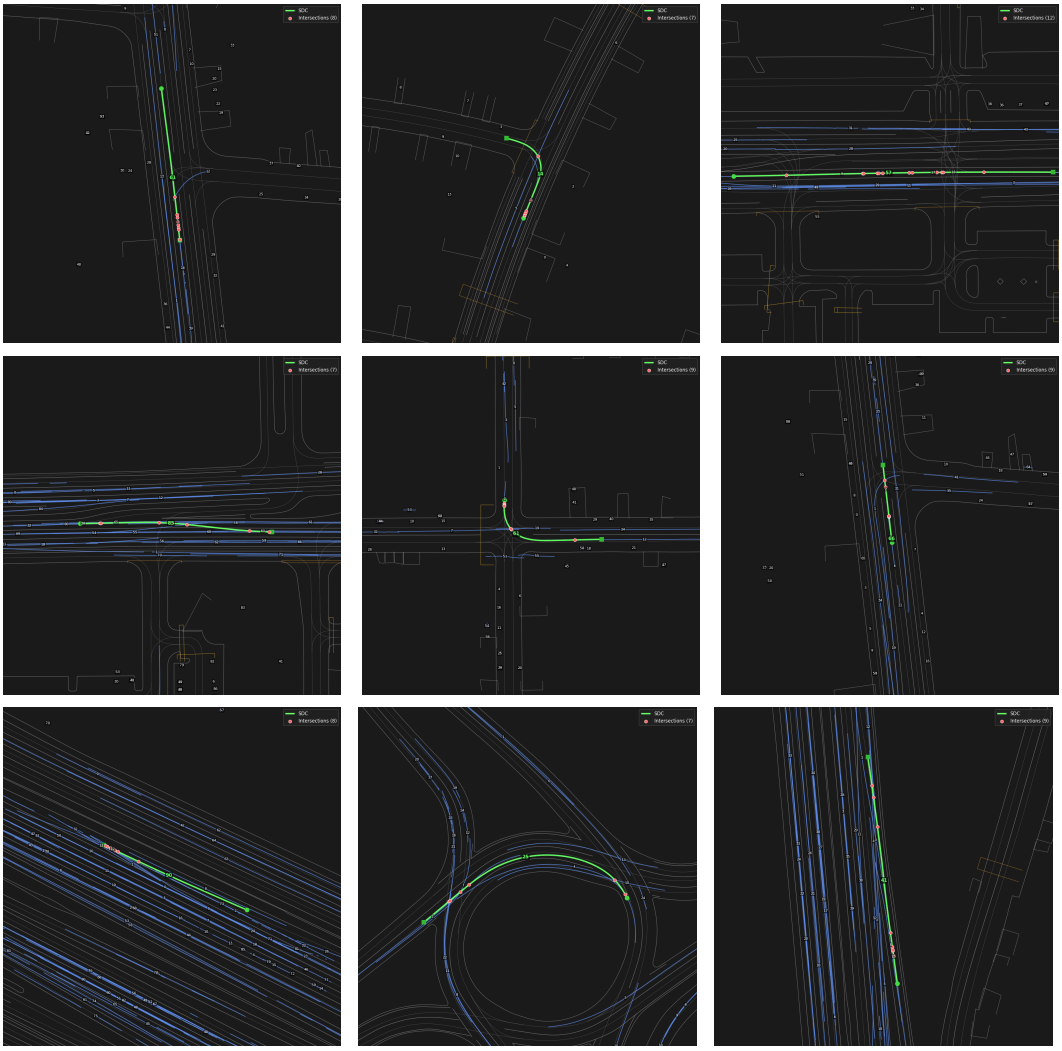

Figure 13: Nine example scenarios from the selected interactive subset. The SDC trajectory is shown in green, other agents in blue, and trajectory intersection points with other logs in red.

D.3 Metrics

We report the following metrics across all experiments. Unless noted, all metrics are computed per active (i.e., controlled) agent per episode and averaged across agents and scenarios.

Score. An agent scores 1 if it reaches its goal without any collision or off-road event during the episode, and 0 otherwise. It jointly captures all failure modes and is a useful aggregate metric.

Completion rate. The fraction of agents that reach their goal position (within $\delta = 2$ meters) before episode end, regardless of whether a collision or off-road event occurred.

Collision rate. The fraction of episodes in which the agent is involved in at least one collision with another vehicle.

At-fault collision rate. A subset of the collision rate taken from NAVSIM [64]. A collision is attributed to an agent if (i) the other vehicle is in front of the agent at the time of impact, and (ii) the

agent's velocity vector points toward the other vehicle. This filters out collisions in which the agent was rear-ended or struck laterally by an inattentive partner.

Collision severity (Δv). Beyond the binary collision indicator, we measure the severity of each at-fault collision event using the change in velocity (Δv) imparted to the agent at impact. Following the impulse-momentum formulation used in [27], the Delta-V of agent i in a collision with partner j is

$$\Delta v_i = \frac{m_j}{m_i + m_j} (1 + e) \left(\vec{v}_j - \vec{v}_i \right) \cdot \hat{n}, \tag{11}$$

where $\hat{n}$ is the unit collision normal (taken as the vector from agent i's center to agent j's center at impact), $e = 0.1$ is the coefficient of restitution for vehicle-to-vehicle crashes, and the dot product is clipped at zero to ignore separating velocities. Masses are proxied from bounding-box footprint for vehicles (anchored at $1500\,\mathrm{kg}$ for a $4.5\,\mathrm{m} \times 1.8\,\mathrm{m}$ reference sedan) and fixed for vulnerable road users ($75\,\mathrm{kg}$ for pedestrians, $90\,\mathrm{kg}$ for cyclists). Δv is one of the strongest predictors of injury risk in vehicle-to-vehicle crashes [27] and lets us distinguish low-impact contacts (e.g. parking-lot taps) from high-energy collisions even when the binary collision rate is identical.

Off-road rate. The fraction of episodes in which the agent crosses a road edge boundary, detected by checking for intersection between the agent bounding box and any road edge polyline.

Route progress ratio. Following [66], we measure how far along its expert reference trajectory each agent travels. At each timestep t, we find the closest point $x(t)$ on the agent's logged trajectory and compute its arc-length distance $d_{x(t)}$ from the start of the path. The route progress ratio is

$$\rho = \frac{d_{x(t)} - d_p}{d_q - d_p}, \tag{12}$$

where d_p and d_q are the arc-length distances to the initial and final positions of the logged trajectory, respectively. A value of $\rho = 1$ means the agent reached its destination; $\rho > 1$ is possible if the agent overshoots. For agents that reach their goal under GOAL_REMOVE termination, we set $\rho = 1$ directly, since their position is invalidated upon removal. For all other agents, ρ is computed from the agent's final position at episode end.

Lateral deviation. At each timestep t, we compute the Euclidean distance from the agent's current position to the nearest point on its expert reference trajectory. We average this distance over all timesteps for which the agent is alive, yielding a mean lateral deviation in meters. This measures how far the agent drifts sideways from its intended route; a perfect replay would achieve zero lateral deviation.

E Additional Results

E.1 Human driving data

E.2 Metadata scaling

The results in Section 4.1 used the best policies across every category, trained within 50k scenarios and focused on scaling the amount of human *driving* data. Next to driving data, a second and distinct source of data is scenario *metadata*: road graphs, initial agent positions, and velocities. Metadata is typically cheaper to obtain than human driving logs and ships with most available driving datasets [24, 67]. Recent work has shown that regularized self-play RL grounded by target-city metadata can adapt driving policies to new cities [23]. A natural follow-up question is how much the diversity provided by metadata matters for training generalizable policies, which is what we explore here.

To do this, we train regularized and vanilla self-play RL agents on nested subsets $\mathcal{M}_k$ with $|\mathcal{M}_k| \in \{10, 100, 1{,}000, 10{,}000, 50{,}000\}$ scenarios, holding the BC anchors τ^n and reward function r fixed. This isolates the effect of *scenario diversity*. Each policy is trained for 20 billion steps.

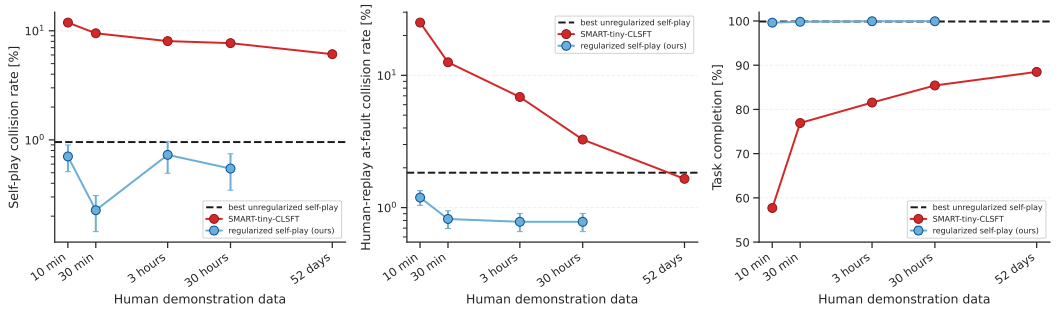

Figure 14: Scaling human driving data for reg. self-play RL; Same as Figure 2 but with the collision rates on a log scale.

As shown in Figure 15, both vanilla and regularized self-play improve drastically with the amount of metadata. For vanilla self-play, the at-fault collision rate drops from 14% at 10 scenarios to 0.5-1% at 50k scenarios, and the human-replay collision rate falls from 25.2% to 2% over the same range. Regularized self-play follows the same trend and reaches lower absolute values: with a 30-min BC anchor, the human-replay at-fault collision rate drops from 14% at 10 scenarios to 0.7% at 50k scenarios. The Zero-Shot Coordination (ZSC) gap approaches 0 for regularized policies, and is 1.9% for vanilla self-play.

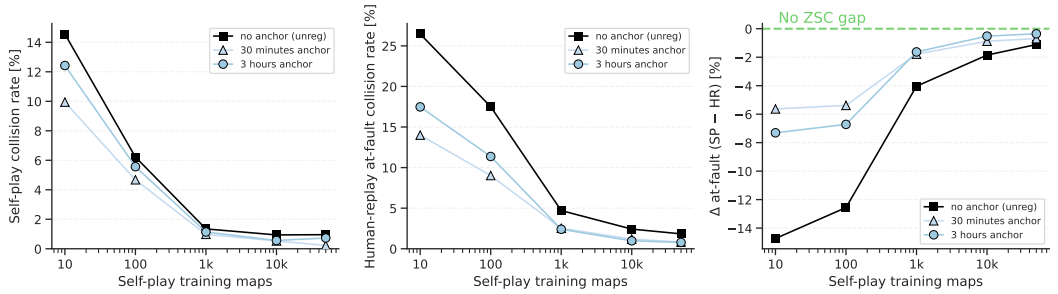

Figure 15: **Scaling laws for scenario metadata**. The unregularized self-play baseline is shown in black; shades of blue correspond to regularized policies trained with different BC anchors, with darker shades indicating more anchor data. *Left:* collision rate in self-play, where all agents are controlled by the same policy on a held-out validation set. *Center:* at-fault collision rate, the fraction of collisions caused by the controlled agent. This metric is more informative than raw collision rate, because replay agents frequently strike the SDC from behind, and such collisions are unlikely to occur in practice (See cartoon in Figure 11). *Right:* ZSC gap Δ_{ZSC}, the difference in the at-fault collision rate between the self-play and human-replay settings.

E.3 Regularization keeps RL policies close to human anchors

Figure 16 shows task completion and KL divergence to the anchor policy over training. Both regularized and unregularized agents converge to comparably effective strategies in terms of goal completion and collision avoidance, yet the underlying action distributions diverge substantially. Without regularization, the agent drifts freely through the space of competent policies, converging far from human behavior; KL divergence increases monotonically throughout training. Regularization constrains the trajectory through policy space without restricting the set of achievable outcomes: the agent remains free to discover effective strategies, but the penalty keeps those strategies within the behavioral distribution of human driving. The result is an agent that is both capable and closer to the distribution of human driving.

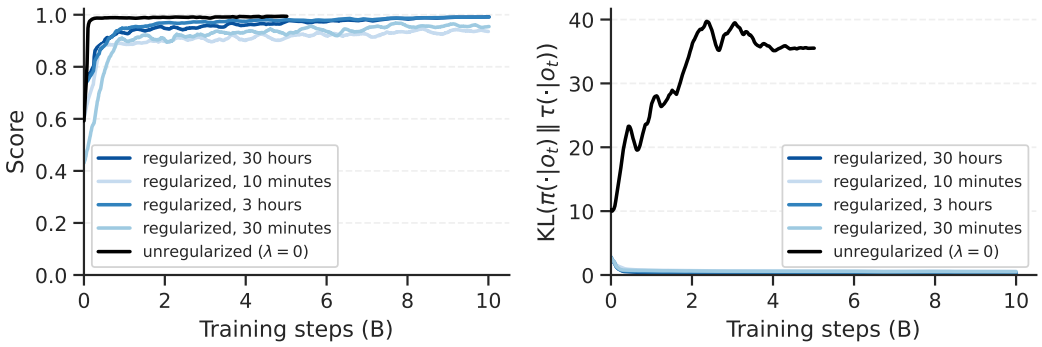

Figure 16: **Regularized self-play remains close to the anchor distribution while unregularized self-play diverges.** Both agents converge to effective driving strategies (left), but their action distributions differ, as measured by KL divergence between observation-conditioned action distributions (right). Regularized policies stay near the anchor; unregularized policies diverge monotonically.

E.4 Safety analysis

Table 8: Collision severity tail breakdown with human-replays in interactive held-out scenarios. *Events* shows the count and share of all collision events attributed to each group. Per-event Δv statistics and the fraction of events exceeding three injury-risk thresholds (1 mph: cosmetic; 5 mph: airbag-deployment floor; 15 mph: elevated serious-injury risk). Best value per column in **bold**; lower is better throughout.

Method	Events (at-fault coll. rate)	Mean Δv (m/s) ↓	Max Δv (m/s) ↓	>1 mph (%) ↓	>5 mph (%) ↓	>15 mph (%) ↓
unregularized	91 (5.0%)	2.09	13.71	**89.0**	54.9	14.3
regularized	53 (2.8%)	**1.71**	**8.09**	90.6	**54.7**	**7.5**

E.5 Distributional Realism: Waymo Open Sim Agent Challenge

Figure 17 reports the WOSAC [26] realism meta-score alongside its three group metrics (kinematic, interactive, and map-based); Figure 18 breaks down all nine submetrics that together make up the meta-score.

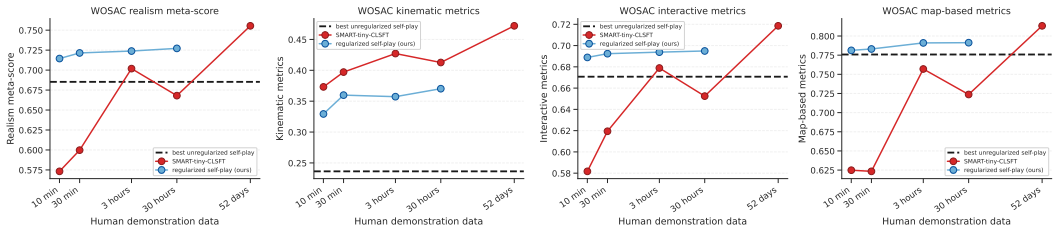

Figure 17: WOSAC meta-scores and group metrics.

Vanilla self-play. Unregularized self-play achieves a WOSAC meta-score of 0.68, with the largest deficits in the kinematic (0.22) and interactive groups. As shown in Figure 18, these policies produce low likelihoods particularly in linear speed, acceleration, and distance to nearest object.

Regularized self-play. Adding regularization improves the meta-score to 0.725, with gains over vanilla self-play across every metric. The score is largely insensitive to additional data.

SMART-tiny CLSFT. SMART trained on 52 days of human data achieves the highest meta-score of 0.755, despite a worse collision rate and task completion across all data bins (Table 1). This result is consistent with SMART's training objective: by optimizing directly for the log-likelihood of recorded trajectories, the model is explicitly supervised to replicate the kinematic and behavioral distributions present in human driving, which WOSAC's realism metrics measure directly.

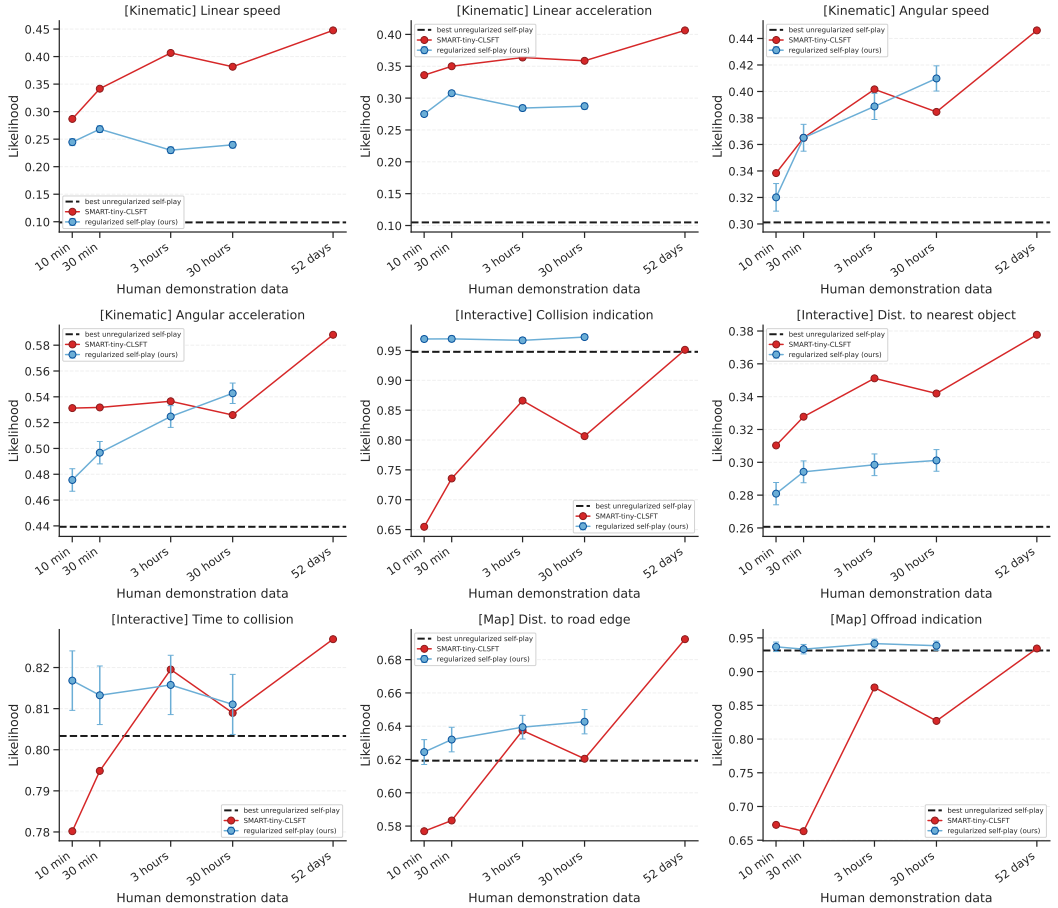

Figure 18: WOSAC submetrics

E.6 Single and multi-agent RL

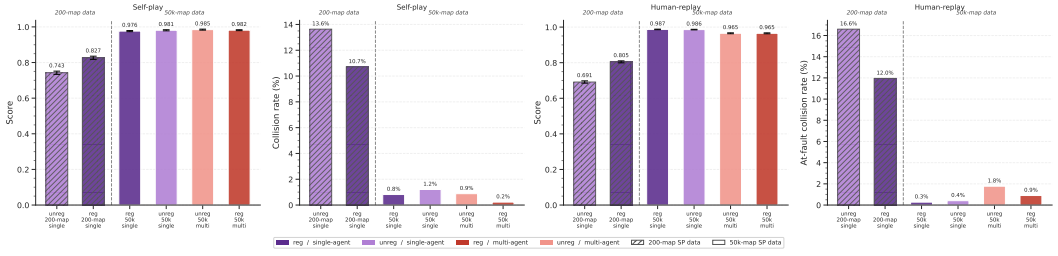

Figure 19: Single vs. multi-agent experiments

E.7 Qualitative analysis

Regularized agents achieve substantially lower longitudinal L2, lateral L2, and displacement errors, reflecting closer alignment with how humans would navigate the interactive traffic scenes. The training curves in Figure 16 provide some grounding for the above results: Regularized self-play agents converge to similar performance as vanilla self-play, but, in contrast to vanilla self-play, the KL divergence to the anchor policy remains small throughout training.

Table 9: Main results comparing unregularized and regularized self-play at 50k training maps. Self-play score on 10k validation scenes; human-replay and IDM-replay metrics on 200 interactive validation scenes. Best value per column in **bold**.

	Score			Human-replay (interactive)			
Method	Self-play	HR	IDM	At-fault (%) ↓	Long. L2 ↓	Lat. L2 ↓	ADE ↓
Unregularized	$\mathbf{0.986 \pm 0.002}$	0.908 ± 0.006	$\mathbf{0.893 \pm 0.007}$	4.9 ± 0.5	13.327 ± 0.129	2.390 ± 0.148	14.074 ± 0.182
Regularized (ours)	0.975 ± 0.002	$\mathbf{0.931 \pm 0.006}$	0.890 ± 0.007	$\mathbf{2.6 \pm 0.4}$	$\mathbf{5.559 \pm 0.077}$	$\mathbf{1.274 \pm 0.029}$	$\mathbf{5.927 \pm 0.076}$

F Extended limitations

Failure modes and directions for improvement. We perform an additional analysis to better understand the limitations of the resulting regularized policies. To improve the signal of the analysis, we evaluate on a curated set of *interactive* scenarios within the held-out set, that is, filter for scenarios that contain dense multi-agent interactions such as merges, unprotected turns, and yielding (details in Appendix D.2).

Table 10 shows that (at-fault) collision rates increase noticeably in these interactive scenarios, even for the best regularized policy (2.1-2.8%) and SMART-tiny-CLSFT trained on 52 days of data (2.7%). We also share several representative failure modes on the webpage `https://sites.google.com/view/anonymous-human-like-autonomy/` (see failure modes).

One possible reason for the increased collision rates for the self-play policies is that the WOMD scenarios that we train in during self-play are small (constructed from a 9-second log), and agent interactions are relatively sparse (see Figure 12 for the distribution of intersections between agent logs), so the RL agent only occasionally trains on transitions that improve difficult coordination situations.

We outline several directions for future work that could improve robustness:

1. **Curriculum learning based on advantage.** Each scenario can be treated as a level whose difficulty is measured by the agent's average advantage. Upsampling scenarios proportionally to their advantage would concentrate training signal on cases the agent finds difficult, naturally increasing exposure to rare but safety-critical situations such as sudden cut-ins and stationary obstacles.
2. **Domain randomization.** Masking out the observation of a ratio of agents within each scenario ("blind" agents [5]) and adding noise to the dynamics or partner features provides a targeted form of domain randomization that could make policy behavior more cautious.
3. **Adversarial fine-tuning.** A third training stage that fine-tunes on a curated set of adversarial human data would expose the policy to scenarios where the other agents in the scene do not respond to it.
4. **Human-like opponents.** Occasionally replacing the self-play opponent with the BC anchor rather than a copy of the RL policy would expose the agent to more human-like partner behavior throughout training.
5. **Stronger anchor policy.** The BC anchor is itself a limiting factor: our best anchor achieves a closed-loop score of 0.66 (Table 6), and a stronger anchor, whether through architectural improvements or additional data, would give the KL regularizer a more reliable behavioral target.

Table 10: Interactive evaluation across all scaling checkpoints. All metrics are computed on the interactive validation subset; policies are rolled out in each of the 200 scenarios 10 times. Top-3 values per column are highlighted (best , 2nd , 3rd); best value additionally in bold. Gray marks the best unregularized self-play value per column. IDM results are not available for SMART (indicated by —).

Self-play maps (metadata)	Anchor data (human demos)	Score		Collision rates			
		HR Score ↑	IDM Score ↑	IDM At-fault (%) ↓	HR At-fault (%) ↓	IDM Coll. (%) ↓	HR Coll. (%) ↓
10	0 (unreg.)	0.312 ± 0.010	0.296 ± 0.010	42.8 ± 1.1	46.2 ± 1.1	46.6 ± 1.1	50.1 ± 1.1
100	0 (unreg.)	0.598 ± 0.011	0.577 ± 0.011	28.9 ± 1.0	29.9 ± 1.0	34.6 ± 1.1	34.3 ± 1.0
1k	0 (unreg.)	0.868 ± 0.007	0.842 ± 0.008	5.8 ± 0.5	7.6 ± 0.6	10.1 ± 0.7	12.2 ± 0.7
10k	0 (unreg.)	0.891 ± 0.007	0.876 ± 0.007	3.2 ± 0.4	4.1 ± 0.4	9.0 ± 0.6	10.2 ± 0.7
50k	0 (unreg.)	0.908 ± 0.006	0.893 ± 0.007	3.8 ± 0.4	4.9 ± 0.5	7.6 ± 0.6	8.7 ± 0.6
10	30 minutes	0.425 ± 0.011	0.432 ± 0.011	33.1 ± 1.0	34.6 ± 1.1	36.6 ± 1.1	37.6 ± 1.1
10	3 hours	0.361 ± 0.011	0.371 ± 0.011	37.3 ± 1.1	39.6 ± 1.1	39.8 ± 1.1	43.2 ± 1.1
100	30 minutes	0.722 ± 0.010	0.661 ± 0.010	16.8 ± 0.8	18.0 ± 0.8	22.4 ± 0.9	23.6 ± 0.9
100	3 hours	0.658 ± 0.010	0.629 ± 0.011	21.8 ± 0.9	24.0 ± 0.9	25.5 ± 1.0	28.2 ± 1.0
1k	30 minutes	0.897 ± 0.007	0.858 ± 0.008	4.4 ± 0.5	5.9 ± 0.5	8.4 ± 0.6	9.2 ± 0.6
1k	3 hours	0.886 ± 0.007	0.866 ± 0.008	5.3 ± 0.5	7.0 ± 0.6	9.3 ± 0.6	10.2 ± 0.7
10k	10 minutes	0.916 ± 0.006	0.858 ± 0.008	3.1 ± 0.4	3.0 ± 0.4	8.3 ± 0.6	6.8 ± 0.6
10k	30 minutes	0.926 ± 0.006	0.892 ± 0.007	3.5 ± 0.4	2.4 ± 0.3	7.9 ± 0.6	7.1 ± 0.6
10k	3 hours	0.906 ± 0.006	0.873 ± 0.007	3.0 ± 0.4	3.5 ± 0.4	7.7 ± 0.6	7.9 ± 0.6
10k	30 hours	0.925 ± 0.006	0.904 ± 0.007	2.6 ± 0.4	3.5 ± 0.4	5.9 ± 0.5	6.0 ± 0.5
50k	10 minutes	0.923 ± 0.006	0.883 ± 0.007	3.1 ± 0.4	3.0 ± 0.4	7.4 ± 0.6	6.9 ± 0.6
50k	30 minutes	0.931 ± 0.006	0.890 ± 0.007	2.8 ± 0.4	2.6 ± 0.4	5.6 ± 0.5	6.0 ± 0.5
50k	3 hours	0.935 ± 0.005	0.890 ± 0.007	3.6 ± 0.4	2.8 ± 0.4	6.5 ± 0.5	5.2 ± 0.5
50k	30 hours	$\mathbf{0.949 \pm 0.005}$	$\mathbf{0.908 \pm 0.006}$	$\mathbf{2.2 \pm 0.3}$	$\mathbf{2.1 \pm 0.3}$	$\mathbf{5.2 \pm 0.5}$	$\mathbf{4.2 \pm 0.4}$
—	10 min (SMART)	0.048 ± 0.005	—	—	35.0 ± 1.1	—	43.9 ± 1.1
—	30 min (SMART)	0.148 ± 0.008	—	—	24.5 ± 1.0	—	30.9 ± 1.0
—	3 hours (SMART)	0.319 ± 0.010	—	—	15.3 ± 0.8	—	21.2 ± 0.9
—	30 hours (SMART)	0.376 ± 0.011	—	—	6.4 ± 0.5	—	11.6 ± 0.7
—	52 days BC (SMART)	0.383 ± 0.011	—	—	4.5 ± 0.5	—	7.9 ± 0.6
—	52 days CLSFT (SMART)	0.433 ± 0.011	—	—	2.7 ± 0.4	—	5.4 ± 0.5

G Mapping Agent Experience to Human Time

We train self-play RL agents on 20 billion transitions. Since Waymo scenarios are discretized at 10 Hz, each transition (o_t, a_t) corresponds to 0.1 seconds of real time, placing the total training experience at approximately 63 years of driving.

For comparison, SMART [11] was trained on the full Waymo Open Motion Dataset, which contains 500,000 training scenarios. Each scenario contributes roughly 90 transitions of SDC trajectory data at 10 Hz, amounting to approximately 45 million transitions in total. The open-sourced SMART-CLSFT checkpoint was trained on all agents in each scene rather than the SDC alone; assuming an average of 5 agents per scenario, this corresponds to roughly 225 million transitions.

Our own checkpoints are trained on subsets of 67, 200, 1,200, and 12,000 maps, each contributing approximately 90 transitions per scenario.

2,500 x claim in the abstract comes from 200 scenarios $\times$ 9 seconds each $=$ 30 minutes. 500,000 $\times$ 9 seconds = 75,000 minutes. 30 minutes / 75,000 minutes = 0.0004.