Reinforcement Learning (CMPE-260)

End-to-End Autonomous Driving using Proximal Policy Optimization

Project Check-in 2

Group-05

Niharika Janapati - 017553744
Sai Harshitha Peddi - 018222087
Bhargava Phaneendra Kumar Suryadevara – 018176379
Venkata Satya Charan Naga Deep -019127238

Repository
[End-to-End Autonomous Driving using Proximal Policy]

Date: November 06, 2025

Abstract

Autonomous driving couples perception, decision-making, and control in safety-critical settings. Training directly in the real world is risky and expensive; simulation offers a scalable alternative for experimentation and iterative improvement. We study end-to-end reinforcement learning for urban driving using Proximal Policy Optimization (PPO) [1] in the CARLA simulator [2]. Our policy learns continuous control (steering, throttle, brake) from monocular RGB input via a shared convolutional encoder and actor–critic heads, optimized with clipped objectives, generalized advantage estimation (GAE), and entropy regularization for exploration [1]. To improve generalization, we adopt curriculum learning (progressing from simple to complex traffic/weather) and domain randomization (lighting, precipitation, traffic actors), addressing known distribution-shift failures in imitation-based methods [4], [5]. We evaluate on standardized CARLA routes and the NoCrash benchmark, reporting route completion, collision rate, and lane deviation, with ablations on representation (raw images vs. compressed latents) and training strategies. Results (to be completed) are expected to show PPO's stability and competitiveness as an end-to-end baseline, and to yield insights into design choices (rewards, curricula, observation encoding) for robust autonomous navigation.

Introduction

End-to-end learning maps raw sensory inputs directly to control, avoiding brittle modular pipelines. Early results showed feasibility with convolutional networks predicting steering from camera frames [3]. However, real-world training by trial-and-error is not viable at scale; realistic simulation (CARLA) provides controllable environments, repeatable scenarios, and safe failure modes [2]. Among modern RL algorithms, PPO [1] is widely adopted for continuous control due to its implementation simplicity and learning stability. Our goal is to implement a transparent, from-scratch PPO agent for CARLA, add principled robustness tactics (curriculum + randomization), and evaluate on established benchmarks to quantify progress beyond pure imitation learning [4]– [6].

Contributions. (i) A clean PPO implementation tailored to CARLA's continuous control; (ii) a training regimen combining curriculum learning and domain randomization; (iii) a systematic evaluation on NoCrash and standardized CARLA routes with metrics aligned to safety and comfort; (iv) ablations isolating the effects of rewards, curricula, and visual representations.

Background and Related Work

End-to-end imitation learning. Bojarski et al. trained a CNN to predict steering from monocular RGB, demonstrating sensorimotor control without explicit perception modules [3]. To handle multi-modal decisions at intersections, Conditional Imitation Learning (CIL) incorporated high-level commands into the policy input, improving route compliance [4]. Nevertheless, behavior cloning suffers from covariate shift: small prediction errors amplify when the vehicle drifts off-distribution. The NoCrash benchmark formalized challenging scenarios (dynamic traffic, adverse weather), revealing generalization gaps for imitation-only systems [5].

Bridging the perception–control gap. Learning by Cheating (LBC) trained a privileged expert using ground-truth maps/semantics, then distilled to a vision-only student, substantially reducing infractions in CARLA [6]. Parallel efforts explored sensor fusion (e.g., TransFuser) that combines LiDAR and camera streams via transformers, yielding safer navigation under occlusion and clutter [7]. These methods highlight the value of privileged information or multi-modal cues—but they increase system complexity and data requirements.

Reinforcement learning for driving. Classical policy-gradient and actor–critic methods (A3C, DDPG) enabled continuous control from pixels but were sensitive to hyperparameters and prone to instability [8], [9]. SAC introduced maximum-entropy objectives to improve exploration and sample efficiency [10]. PPO stabilized on-policy learning via clipped policy updates, striking a practical balance between stability and performance, which motivated its adoption here [1]. Compared with imitation learning, RL naturally optimizes task-level objectives (safety, comfort) through reward design and can incorporate curricula and randomization to improve robustness to shift [5].

Positioning. Our work investigates a research-clean PPO baseline in CARLA—emphasizing transparent implementation, robust training curricula, and rigorous benchmarking. While leaderboard-topping systems often rely on fusion and/or privileged perception [6], [7], a well-tuned end-to-end PPO agent remains a valuable reference point for understanding the trade-offs among stability, generalization, and simplicity.

Problem Formulation

We consider a partially observed Markov decision process (POMDP) with observation $o_t \in \mathbb{R}^{H\times W\times 3}$(RGB image), continuous action $a_t = [\delta_t, \tau_t, \beta_t]$(steer, throttle, brake), reward r_t, and environment dynamics given by CARLA [2]. The objective is to maximize the expected discounted return $\mathbb{E}[\sum_t \gamma^t r_t]$while respecting road rules and safety. We measure performance via route completion, collision count, and lane deviation; secondary metrics include **control smoothness** (jerk) and average reward per episode.

Methodology

Observation & Action Spaces:

Observations are monocular RGB images (e.g., 84×84 or 128×128). Optionally, we compare compressed latent encodings produced by a lightweight autoencoder to study the representation–sample-efficiency trade-off (motivated by findings in LBC [6]). Actions are continuous: steering $\delta \in [-1,1]$, throttle $\tau \in [0,1]$, brake $\beta \in [0,1]$.

Network Architecture:

A shared convolutional encoder (Conv-BN-ReLU blocks) feeds two MLP heads: the **actor** outputs a Gaussian policy (mean, log-std) over actions; the critic predicts state value $V(o_t)$. We use orthogonal initialization, layer-norm where helpful, and tanh squashing for stable control. The encoder capacity is tuned to balance sample efficiency and inference latency.

PPO Objective and Optimization:

We implement clipped PPO with minibatch SGD over trajectory rollouts:

$$\mathcal{L}_{\text{PPO}} = \mathbb{E}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] - \lambda_V(V_\theta - \hat{R}_t)^2 + \lambda_H \mathcal{H}(\pi_\theta(\cdot \mid o_t)),$$

where $r_t(\theta)$is the probability ratio, $\hat{A}_t$is GAE advantage, $\epsilon$is the clip parameter, and $\lambda_H$weights entropy regularization [1]. We apply reward normalization, value clipping, gradient clipping, and a cosine LR schedule.

Reward Design:

We combine: (i) progress along route, (ii) lane keeping (negative penalty for lateral deviation), (iii) speed regularization (encouraging target speed), (iv) infraction penalties (collisions, off-road, red-light), and (v) small action-smoothness penalties. We calibrate weights using short pilot runs, following findings that well-shaped rewards curb unsafe exploration and reduce sample complexity [4], [5].

Robustness: Curriculum & Domain Randomization:

We start in Town01 with low traffic and clear weather; as performance stabilizes, we add intersections, pedestrians, and adverse weather (rain, night), then domain-randomize weather, lighting, and traffic patterns per episode. This strategy explicitly targets the generalization failures highlighted by NoCrash [5].

Baselines & Ablations:

Baselines include: (i) imitation-learning (CIL-style supervised policy) [4]; (ii) A3C/DDPG RL baselines [8], [9]. Ablations: (a) reward components, (b) curriculum on/off, (c) randomization on/off, (d) raw images vs. latent encoder (inspired by [6]).

Data and Experimental Setup

Benchmarks and Datasets:

We use CARLA's simulated environments and three community-standard resources for training/evaluation:

- CARLA Autonomous Driving Dataset (CARLA AD) — RGB, segmentation labels, control signals for supervised validation and encoder pretraining. CARLA AD
- NoCrash Benchmark — predefined urban routes with varying traffic and weather for robustness testing [5]. NoCrash Benchmark
- CARLA Leaderboard Evaluation Suite — standardized routes and scoring for fair comparison across methods. CARLA Leaderboard

Simulator Configuration:

We use CARLA 0.9.x [2] with synchronous stepping and fixed-delta physics for determinism during training. Sensors: one forward RGB camera (60–90° FOV, 10–20 Hz). Traffic manager spawns vehicles/pedestrians per scenario. Towns: Town01, Town03, Town05 for train/val/test splits to measure cross-town generalization.

Data Generation & Logging:

Custom Python drivers collect (o, a, r, o′, done) tuples. We log per-step events (collisions, lane-invasions), weather seeds, and route IDs for reproducibility. Each configuration generates ~50 hours of simulated driving across seeds, stored as shards with PNG frames and JSON/NPZ metadata to enable offline analysis and replays.

Preprocessing & Batching:

RGB frames are resized to 84×84 (or 128×128 for ablations), normalized to [0,1], and optionally frame-stacked (k=3–4) to encode short-term dynamics. Lightweight augmentations (brightness/contrast jitter, Gaussian noise) emulate camera variability. Training uses PyTorch DataLoaders with pinned memory and mixed precision (FP16) on GPU.

Training Protocol:

We adopt rollout lengths of 128–256, PPO epochs 3–10, minibatch size 32–128, discount $\gamma = 0.99$, GAE $\lambda = 0.95$, entropy coeff 0.01–0.02, and clip $\epsilon = 0.1$–0.2[1]. Curriculum stages: (S1) clear-day, low traffic; (S2) variable weather; (S3) intersections and pedestrians; (S4) heavy traffic + full randomization. We checkpoint every N updates and select the best model by validation route completion.

Evaluation Metrics & Protocol:

Primary metrics: Route Completion (%), Collisions per km, Avg. Lane Deviation (m). Secondary: Longitudinal/Lateral Jerk (comfort), Avg. Episode Reward. We report mean ± std over ≥3 seeds. On NoCrash, we follow its town/weather splits [5]; on Leaderboard, we report official scores. Qualitative analysis includes trajectory overlays and failure taxonomies (e.g., occlusions, yielding errors).

Compute & Reproducibility:

Training runs on SJSU HPC GPUs (e.g., V100/A100). We publish configs, seeds, and scripts in the repo for one-click reproduction. All figures/tables will include seed counts and confidence intervals.

References:

[1] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” arXiv:1707.06347, 2017.

[2] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An Open Urban Driving Simulator,” arXiv:1711.03938, 2017.

[3] M. Bojarski et al., “End to End Learning for Self-Driving Cars,” arXiv:1604.07316, 2016.

[4] F. Codevilla, M. Müller, A. Lopez, V. Koltun, and A. Dosovitskiy, “End-to-End Driving via Conditional Imitation Learning,” in Proc. IEEE ICRA, 2018.

[5] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, “Exploring the Limitations of Behavior Cloning for Autonomous Driving,” in Proc. ICCV, 2019, pp. 9329–9338.

[6] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, “Learning by Cheating,” in Proc. CoRL, 2019.

[7] A. Prakash, K. Chitta, and A. Geiger, “Multi-Modal Fusion Transformer for End-to-End Autonomous Driving,” in Proc. CVPR, 2021, pp. 7077–7087.

[8] V. Mnih et al., “Asynchronous Methods for Deep Reinforcement Learning,” in Proc. ICML, 2016, pp. 1928–1937.

[9] T. P. Lillicrap et al., “Continuous Control with Deep Reinforcement Learning,” arXiv:1509.02971, 2015.

[10] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,” in Proc. ICML, 2018, pp. 1861–1870.