Supplementary material – Learning to drive from a world on rails

Dian Chen
UT Austin

Vladlen Koltun
Intel Labs

Philipp Krähenbühl
UT Austin

1. Kinematic Bicyle Model

The kinematics of the bicycle model [7] $\mathcal{T}^{ego}$ used in our CARLA experiment is described below:

$$\begin{aligned}\dot{x} &= v\cos(\theta+\beta)\\ \dot{y} &= v\sin(\theta+\beta)\\ \dot{v} &= a\\ \dot{\theta} &= \frac{v}{r_b}\sin(\beta)\\ \tan(\beta) &= \frac{r_b}{f_b+r_b}\tan(\phi)\end{aligned}$$

We train $\mathcal{T}^{ego}$ in an auto-regressive manner using L1 loss and stochastic gradient descent:

$$\begin{aligned}J_{ego} = &\sum_t^T |x_t-\hat{x}_t| + |y_t-\hat{y}_t|\\ &+\sum_t^T |\cos(\theta_t)-\cos(\hat{\theta}_t)| + |\sin(\theta_t)-\sin(\hat{\theta}_t)|\end{aligned}$$

where $x_{t+1}, y_{t+1}, \theta_{t+1}, v_{t+1} = \mathcal{T}^{ego}(x_t, y_t, \theta_t, v_t, a_t)$, and $a_t = (s_t, t_t, b_t)$. We only model θ as a transform of s; a as a transform of (t, b), and vehicle wheelbases r_b, f_b. We use an action repeat of 5 frames, hence both data collection and planning operate at 4 FPS, whereas the simulator and the visuomotor policy run at 20 FPS.

2. ProcGen Training Levels Returns

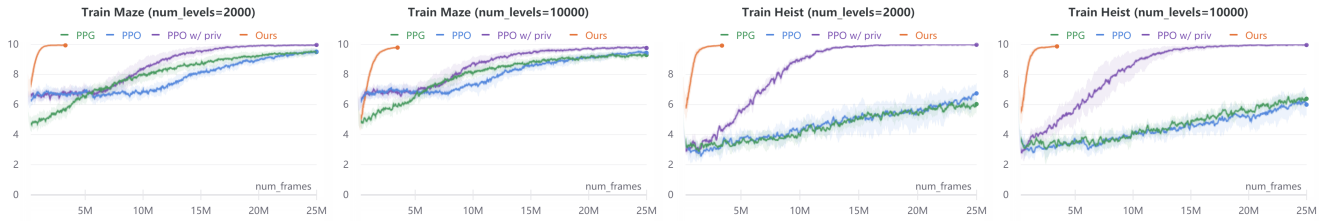
Figure 1 plots the average episode returns of our method against PPO [8], PPG [2], and PPO with access to privileged information.

3. Additional details for CARLA experiments

Dataset. For the CARLA leaderboard, we collect $1M$ frames, corresponding to roughly 69 hours of driving. For the NoCrash benchmark [4], we collect $270K$ frames. The dataset uses a privileged autopilot π_b. However, we do not store the controls from the ego-vehicle autopilot, unlike imitation learning. The RGB image is collected and stitched from three front-facing cameras all mounted at x=1.5m, z=2.4m in the ego-vehicle frame. Each camera has a 60° FOV; the side cameras are angled at 55°. For the CARLA leaderboard, we additionally use a telephoto camera with 50° FOV to capture distant traffic lights. To augment the dataset, we additionally mount two side camera suites with the same setup, each mounted as if the vehicle is angled at $\pm 30^\circ$ following Bojarski et al. [1]. For the CARLA leaderboard, we collect our dataset in the 8 public towns under a variety of weathers. For the NoCrash benchmark, we collect our entire dataset in Town1 under four training weathers, as specified by the CARLA benchmark [5, 3].

Experimental setup. For the CARLA leaderboard, agents are asked to navigate to specified goals through a variety of areas, including freeways, urban scenes, and residential districts, and in a variety of weather conditions. The agents face challenging traffic situations along the route, including lane merging/changing, negotiations, traffic lights, and interactions with pedestrians and cyclists. Agents are evaluated in held-out towns in terms of a Driving Score metric that is determined by route completion and traffic infractions.

In the NoCrash benchmark, agents are asked to safely navigate to specified goals in an urban setting with intersections, traffic lights, pedestrians, and other vehicles in the environment. The NoCrash benchmark consists of three driving conditions, with traffic density ranging from empty to heavily packed with vehicles and pedestrians. Each driving condition has the same set of 50 predefined routes: 25 in the training town (Town1) and 25 in an unseen town (Town2). Agents are evaluated based on their success rates. A trial on a route is considered successful if the agent safely navigates from the starting position to the goal within a certain time limit. The time limit corresponds to the amount of time required to drive the route at a cruising speed of 5 km/h, excluding time spent stopping for traffic lights or other traffic participants. In addition, a trial is considered a failure and aborts if a collision above a preset threshold occurs, or the vehicle deviates from the route by a preset margin. Each trial is evaluated on six weathers, four of which are seen in training and two that are only used at test time. The four training



(a) Maze 2000 training levels (b) Maze 10000 training levels (c) Heist 2000 training levels (d) Heist 10000 training levels

Figure 1: Comparison of our method to state-of-the-art model-free reinforcement learning on the navigational tasks of the ProcGen benchmark. All plots measure the average episode returns on the **training levels**. Experimental setup follows Figure **??**.

weathers are "Clear noon", "Clear noon after rain", "Heavy raining noon", and "Clear sunset". The two test weathers are "Wet sunset" and "Soft rain sunset". We use CARLA 0.9.10 for all experiments.

4. Additional NoCrash Experiments

Table 1 compares our visuomotor agent, which is trained with an auxiliary semantic segmentation loss, with a simpler baseline that does not use this auxiliary loss. Policies trained with semantic segmentation consistently outperform the action-only baseline, especially under generalization settings. We observed the same for the LBC baseline, which also uses semantic segmentation as an auxiliary loss.

Table 2 additionally compares the route completion rates of the presented approach (Ours) to prior state-of-the-art on the CARLA NoCrash benchmark.

Table 3 shows success rates of our method on two additional random seeds in addition to the one in table 2 from main manuscript. Denser traffic results in higher variance in success rates.

Table 4 compares different variation of our visuomotor agent at the distillation stage. **CA** stands for camera augmentation, meaning the model trains on the additional augmented camera images, described in section 4. **SA** stands for "speed augmentation". An **SA** model trains to predict action values on all discretized speed bins, instead of taking as input the recorded speed reading from the dataset. During test time, an **SA** models uses linear interpolation to extract the action-values corresponding to the ego-vehicle speed. Models trained with camera or speed augmentation consistently outperform ones that were not, showing the benefits of dense action-values computed using our factorized Bellman updates. We therefore use camera and speed augmentation for our models for the CARLA leaderboard and the NoCrash benchmark. With the augmented supervision extracted from the dense action-values, models perform well even without trajectory noise injection [6, 4].

		Auxilliary loss	
Town	Weather	×	✓
train	train	95	98
train	test	70	90
test	train	80	94
test	test	46	78

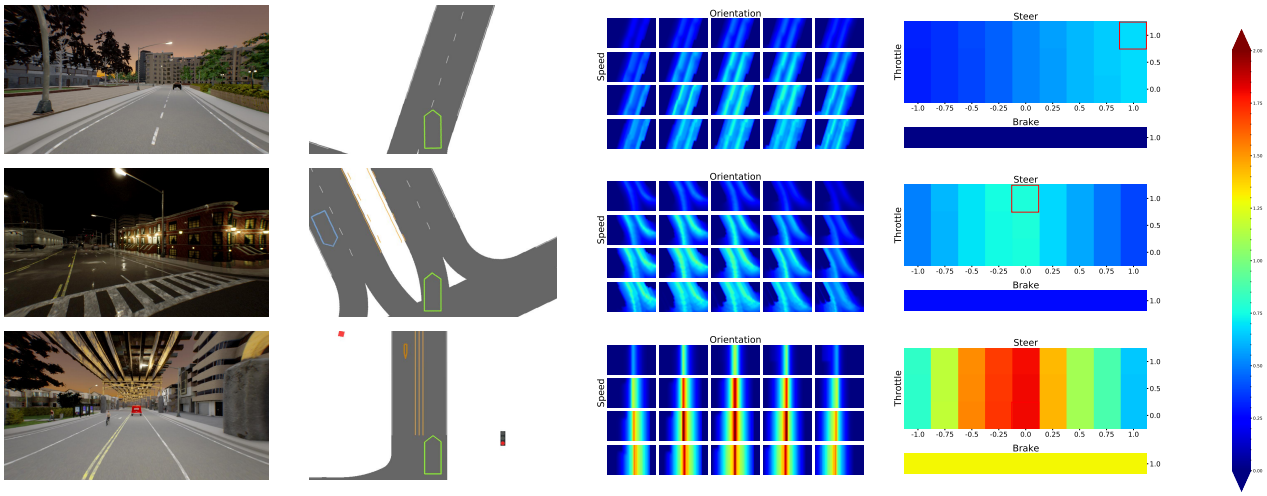
Table 1: Comparison of success rate in the NoCrash benchmark on the empty traffic condition with and without the auxiliary semantic segmentation loss.

Task	Town	Weather	IA	LBC	**Ours**
Empty			95.02	97.15	**98.82**
Regular	train	train	94.72	96.38	**100.00**
Dense			82.93	91.35	**98.24**
Empty			*88.87*	92.41	**98.91**
Regular	test	train	*84.09*	88.32	**94.95**
Dense			*63.63*	74.84	**88.89**
Empty			–	79.35	**94.25**
Regular	train	test	–	79.20	**93.03**
Dense			–	76.72	**95.73**
Empty			–	62.47	**84.72**
Regular	test	test	–	63.55	**88.53**
Dense			–	44.99	**80.75**

Table 2: Comparison of the **mean route completion rate** on NoCrash. The experimental setup follows Table 2.

5. Action-value Computation

In CARLA, we use a planning horizon of $H = 5$ to subsample the trajectories during action-value computation. At each frame t, we compute and discretize the rewards from t to $t + H - 1$ around the ego vehicle state at time t. We then compute the values and action-values for time t using backward induction as described in section 3. In ProcGen, we use $H = 30$.



(a) RGB camera (b) Map (c) Value-map (d) Action-value

Figure 2: Additional visualization of the computed value function and action-value function for the current frame. Setup follows figure 3 in the main manuscript.

Task	Town	Weather	Seed1	Seed2	Seed3
Empty			99	99	98
Regular	train	train	97	99	100
Dense			86	94	96
Empty			93	92	94
Regular	test	train	94	91	89
Dense			74	67	74
Empty			92	84	90
Regular	train	test	92	92	90
Dense			70	80	84
Empty			78	78	78
Regular	test	test	86	82	82
Dense			60	60	66

Table 3: Success rates of our method on 3 evaluation random seeds. Setup follows table 2 in the main manuscript.

6. CARLA Controls

In CARLA, to ensure a smooth control output from the discretized action space, we assume independence between steering and throttle, and use their softmax probabilities to compute smooth steering and throttle values. In particular, the sensorimotor policies predict logits $\log \pi_s \in \mathbb{R}^{N_s}, \log \pi_t \in \mathbb{R}^{N_t}, \log \pi_b \in \mathbb{R}$. During training we model $\log \pi(s, t, \mathbb{I}_b) = (1 - \mathbb{I}_b)(\log \pi_s(s) + \log \pi_t(t)) + \mathbb{I}_b \log \pi_b$.

During testing, we use

$$s = \sum_{c}^{N_s} \pi_s(s_c) s_c$$

$$t = \sum_{c}^{N_t} \pi_t(t_c) s_c$$

$$b = \begin{cases} 1, \pi_b >= t_b \\ 0, \pi_b < t_b \end{cases}$$

We use $t_b = 0.5$ in all our experiments. In addition, we apply a bang-bang controller on throttle, i.e we explictly set the computed throttle to 0 if the vehicle speed exceeds a predefined threshold.

7. CARLA leaderboard

Following Toromanoff et al. [9], we use a 6 model ensemble to obtain a more stable control for our top leaderboard submission.

8. Additional ProcGen Details

Similar to CARLA, we discretize the agent state into $N_H \times N_W$ location bins and N_θ orientation bins. We use $N_H = N_W = 32$, and $N_\theta = 8$. The ConvNet that processes environment features takes as input a cropped 13×13 region around the ego-agent in the original 64×64 RGB observations. The ConvNet features are concatenated with agent orientation to predict the next ego-agent's states under all discrete action commands.

CA	SA	Train town						Test Town					
		Train Weather			Test Weather			Train Weather			Test Weather		
		Empty	Regular	Dense	Empty	Regular	Dense	Empty	Regular	Dense	Empty	Regular	Dense
×	×	87	82	82	60	78	82	85	80	63	68	54	42
×	✓	97	97	92	78	82	80	92	**91**	64	66	72	58
✓	×	**100**	98	90	**92**	**94**	76	90	82	60	78	62	48
✓	✓	98	**100**	**96**	90	90	**84**	**94**	89	**74**	**78**	**82**	**66**

Table 4: Comparison of success rate in the NoCrash benchmark under different ablation conditions. **CA** stands for "camera augmentation" and **SA** stands for "speed augmentation". All ablation models are trained on the same dataset and evaluated on CARLA 0.9.10. **CA** models additionally train on two augmented camera views per dataset frame.

Hyperparameter	CARLA	ProcGen
Batch size	128	128
Learning rate - ego model	1e-2	3e-4
Learnign rate - distillation	3e-4	3e-4
Entropy loss scale (α)	1e-2	1e-2
Segmentation loss scale	5e-2	–

Table 5: Additional hyperparameters.

9. Training Hyperparameters

Table 5 provide a list of training hyperparameters for reference. In our CARLA experiments we use the following image augmentations: Gaussian Blur, Additive Gaussian Noise, Pixel Dropout, Multiply (scaling), Linear Contrast, Grayscale, ElasticTransformation.

References

[1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. 2016. 1

[2] Karl Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *arXiv preprint*, 2020. 1

[3] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *ICRA*, 2018. 1

[4] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *ICCV*, 2019. 1, 2

[5] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017. 1

[6] Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg. Dart: Noise injection for robust imitation learning. In *CoRL*, 2017. 2

[7] P. Polack, F. Altché, B. d'Andréa-Novel, and A. de La Fortelle. The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *IV*, 2017. 1

[8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. In *arXiv preprint*, 2017. 1

[9] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *CVPR*, 2020. 3