

# Sparse Curriculum Reinforcement Learning for End-to-End Driving

Pranav Agarwal<sup>1</sup>, Pierre de Beaucorps<sup>1</sup> and Raoul de Charette<sup>1</sup>

**Abstract**—Deep reinforcement Learning for end-to-end driving is limited by the need of complex reward engineering. Sparse rewards can circumvent this challenge but suffers from long training time and leads to sub-optimal policy. In this work, we explore driving using only goal conditioned sparse rewards and propose a curriculum learning approach for end to end driving using only navigation view maps that benefit from small virtual-to-real domain gap. To address the complexity of multiple driving policies, we learn concurrent individual policies which are selected at inference by a navigation system. We demonstrate the ability of our proposal to generalize on unseen road layout, and to drive longer than in the training.

## I. INTRODUCTION

Recent advancements in deep reinforcement learning (RL) has successfully solved some of the most challenging tasks. Performance of Deep RL in competing with humans in games like Atari [32], AlphaGo [46] and Dota [2] has shown its potential to solve complex decision making problems for long term gains. Considering these results, similar approach has been used to solve real life applications like robotic manipulation [10], [19] and autonomous driving [37], [3]. However, most of the approaches are domain dependent and are not generalizable across different tasks.

While the traditional rule based approach [20] requires complex engineering and computationally extensive search based algorithms [6], [33] to solve challenging tasks like autonomous driving, learning from rewards propose an alternative where the optimal policy is optimized from reward. However, most of the current learning based approach use reward shaping [36], [23], [50], [41] relying on extensive manual supervision and are subsequently prone to human bias [16]. In real life, humans are often rewarded *only* when the task is complete. Sparse reward follows the same analogy and is domain independent, hence easily transferred to new tasks. However, lack of feedback signals makes sparse reward dependent algorithm difficult to train [12].

One solution is to use curriculum learning that breaks a complex task into sub task of gradually increasing complexity. Driving forms a natural curriculum considering that the driving goal increases with distance. However, for a sparse reward setting it may be impossible to train a single policy to maneuver all road conditions. Different road conditions like left turn, right turn or roundabout require completely different skills. We tackle the above problem by designing a multi-policies strategy where individual policies are learned concurrently. At inference, a navigation system handles the

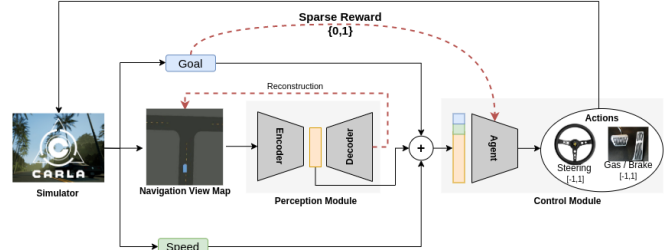


Fig. 1: **End to End Driving.** Overview of our reinforcement learning framework for end to end driving with sparse reward. The perception module encodes the navigation view map and the control module learns the optimal low level control (throttle, steering, brake) from the world model, current speed, distance covered and the goal.

navigation decision at intersections (e.g. turn left), subsequently switching to the ad-hoc policy. As illustrated in Fig. 1, instead of the often used front-view images, we train from navigation view maps and learn a compact world model [11] from the latter. Both world model (VAE) and RL policy (Proximal Policy Algorithm [45]) are trained simultaneously.

The main contribution of our work are:

- **Learning to drive with navigation maps only.** We address RL for driving by learning a compact world model from navigation maps only, having smaller domain gap, and subsequently offering better generalization capacity,
- **Learning individual policies with curriculum and sparse reward.** We use constrained goals of increasing complexity in a curriculum fashion and benefit from our simple revert strategy,
- **Simultaneous perception and control learning.** Our end to end framework can training perception and control modules sequentially or simultaneously which balances the advantage of training policy from pixels and from a compact world model.

## II. RELATED WORKS

We organize our literature review in two fold, covering first end-to-end driving with Reinforcement Learning (RL), and curriculum learning.

### A. End-to-end driving with RL.

Because it requires a trial-and-error process, the common strategy involves training virtual agents with simple rewards, in a model-based RL fashion. Reward shaping is of the utmost importance since it provides a supervision signal

<sup>1</sup>Pranav Agarwal, Pierre de Beaucorps and Raoul de Charette are from Inria, Paris. `first.last-name@inria.fr`

for the network, and was subsequently extensively discussed in [34], [4], [15], [26], [21]. Dense (i.e. frame-wise) reward are commonly applied, by penalizing velocity [21], [25] along with road-car angle [31], distance to center [17], [26], [51], [30], distance from destination [7], car collisions [7]. Some works also optimize comfort accounting for steering and throttle [3] and traffic rules [25]. The main limitation of dense reward is that it acts as an expert to imitate, thus limiting the discovery [28], and preventing natural human-like behavior – like taking a turn from inside [17]. Most RL methods learn directly the pixel-control mapping from front-view cameras [26], [51], [17], [53] though imitation learning also experimented with additional modalities along RGB [13]. A few methods also rely on auxiliary task [52] or intermediate state representation [38] to provide additional supervision and lower the virtual-real gap. Of interest for this work, [38] relates that using front-view camera images limits the transferrability to real world. We refer to the recent survey [48] for details.

Instead, we train using only navigation maps to ease real-world applicability, and employ sparse reward learning thus letting the agent to learn the driving strategy (speeding, lane centering, turning, etc.). Since sparse learning result in long training time we employ curriculum learning reviewed below.

### B. Curriculum learning.

Curriculum is traditionally employed [8], [43], [44] as a way to break down complex tasks into simple ones of increasing complexities – similarly to human learning [22]. It has been reintroduced for deep learning [1], leading to many extensions [56], [29], [9], [39]. On optimization, the intuition is that a simple task can drives the optimizer closer to its minimum [24], [49]. Controlling the complexity is the key element and sophisticated strategies were used here like teacher guided [9], self play [47] or the use of GANs for goal generation [14], [40].

Curriculum learning was shown to be a natural candidate for sparse reward which provides little supervision. Because they avoid the bias of model-guidance, sparse reward was extensively used for robot manipulation [54], [42], [35], [27] or end-to-end flying of Unmanned Aircraft Vehicle (UAV) [55]. We found no prior work on end-to-end driving, which relates possibly to the complexity of using sparse reward in long-horizon tasks [18] or structured environment like road layout.

We address hereafter driving as curriculum, where complexity increases naturally with the goal distance, and providing only sparse binary reward to let the network discover efficient driving strategy.

## III. METHOD

Fig. 1 shows an overview of our reinforcement learning (RL) framework, where the agent interacts with Carla [7] virtual driving environment and predicts the continuous low level control of a car (steering, throttle, brake). There are important originalities of our proposal regarding the literature. Firstly, rather than front view images we rely on navigation views (i.e. top-view navigation maps) which prove smaller

virtual-to-real gap. Secondly, to avoid the pitfall of model-based RL – where agents mimic the model, we only provide a sparse binary reward at the end of each episode ; 1 if the goal is reached, 0 if not. Interestingly, this light supervision, enables the discovery of unique driving features. Thirdly, we learn a compact world model which enables better policy learning, and train the whole pipeline in a curriculum fashion for separate policies (driving straight, turning left, turning right). For inference a simple navigation system enables policy switching.

The remaining of this section describes the architecture (Sec. III-A), our sparse reinforcement learning pipeline (Sec. III-B), and the training strategy (Sec. III-C).

### A. Architecture

Our architecture is composed of two main modules: *perception* which provides a compact representation of the world, and *control* which predicts the full car control output.

The perception module takes as input a single navigation view map (256x256 RGB) output by Carla simulator. It is composed of a shallow Variational AutoEncoder (VAE) having 5 convs with increasing filters (32,64,128,256,256) as encoder, and 5 transpose convs (256,128,64,32,3) in the decoder. The latent space is a 256 dimension vector, hereafter referred as *world model*.

The control module takes as input the concatenation of the world model, with current speed and goal easily retrieved from Carla metadata, as illustrated in Fig. 1. It employs a shallow network consisting of 4 FC layers (512,256,128,64 neurons) and a 2 neuron output layer with tanh activation to output steering and brake/throttle as scalars in  $[-1, +1]$ . The prediction is logically applied to the simulator before obtaining the next simulation step.

### B. Sparse reinforcement learning

Instead of penalizing the agent with dense frame-wise reward, we define a goal to reach and only reward the agent (+1) if goal is reached before the episode termination. As such, our supervision signal is significantly weaker than existing dense reward end-to-end driving [48], but encourages self discovery. As other long-horizon tasks [18], learning driving is indubitably hard with a sparse reward. We employ thus curriculum learning strategy to break complexity, and uses Proximal Policy Optimization (PPO) as it is less sensitive to hyperparameters tuning while ensuring slow policy deviation during training. We first provide the reader with a brief background.

**Proximal Policy Optimization (PPO).** In short, considering a state  $s_t$  and an action space  $a_t$ , PPO [45] optimizes network parameters  $\theta$  using a surrogate objective  $L(\cdot)$  with a stochastic gradient ascent to learn the best policy  $\pi$ , while avoiding large deviation from last policy  $\pi_{\theta_{old}}$ . This writes

$$L(\theta) = E_t \left[ \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} A_t^{\pi_{\theta_{old}}} \right], \quad (1)$$

with  $A_t$  being the advantage of the predicted action – that is, *How good/bad was the predicted action*.

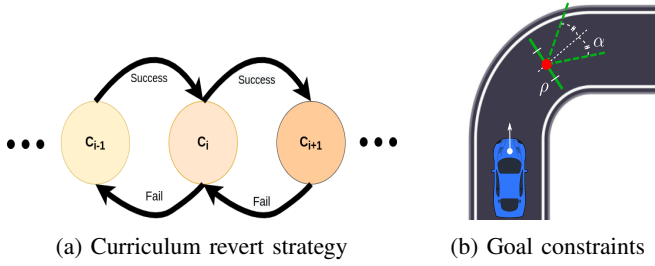


Fig. 2: **Curriculum strategy and Goal constraints.** (a) Instead of simply increasing complexity, we introduce a revert mechanism upon failure. (b) is a schematic view of a goal (red) and our constraints (green). Details in text.

While it was originally proposed with either KL penalty or clipped objective to solve the constrained policy optimization problem, we use the lighter clipped version which writes

$$L^{\text{clip}}(\theta) = E_t [\min(r_t(\theta) A_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) A_t)], \quad (2)$$

with reward  $r_t$  being function of current and old policy,

$$r_t = \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}, \quad (3)$$

bounded by  $\epsilon$  within a range of  $[1 - \epsilon, 1 + \epsilon]$ . This prevents the new policy  $\pi_{\theta}$  from large changes as compared to the old policy  $\pi_{\theta_{old}}$ . To further reduce variance, the generalized advantage estimation  $A_t$  is the weighted average of several advantage functions.

**Curriculum learning.** Unlike dense ones, sparse rewards are known to be *sample inefficient* because of the little guidance provided by the reward. To fight this, we use curriculum learning [1] where *complexity* is the distance the agent needs to drive. We customize the curriculum setup for our problem.

Firstly, instead of monotonically increasing the complexity, we introduce a *revert strategy* – inspired by [5]. While the latter saved checkpoint weights and restored them on-demand, our strategy is to increase complexity only when the agent succeed and decrease it upon failure, as illustrated in Fig. 2a. We argue that similarly to [5], this prevents policy to remain stuck in a local minimum. Empirically, we observe that it plays an important role in the training (see Sec. IV). For each episode, the goal is defined as a 2 DoF (Degree of Freedom) vector being the target position.

Secondly, we apply goal constraints by introducing a lateral radius  $\rho$  to the goal position, and a maximum angle difference  $\alpha$  w.r.t. the local road curvature. Specifically, the goal is being reached when the agent position is within  $[-\rho, +\rho]$  of the goal position *and* if its relative road orientation is  $[-\alpha, +\alpha]$ . The intuition is that it forces the agent to cross a virtual ‘finish line’ with a certain orientation as illustrated in Fig. 2b, logically boosting policy for the next complexity. We set  $\rho = 1m$  and  $\alpha = 15^\circ$ .

In practice, we start the curriculum with a complexity of 1 (i.e. 1m goal) and increment/decrement it by 1 according to our curriculum. Upon goal completion the agent is being

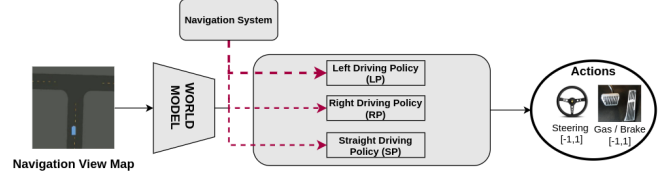


Fig. 3: **Multi-policy strategy.** The navigation system switches policies according to the action required at the next intersection.

rewarded ‘+1’ and the episode terminates. If the goal is not reached before the episode terminates – without any reward. Importantly, we stop the training after 100m goal distance is reached.

### C. Training strategy

Policy is trained as mentioned with a binary sparse reward if goal is completed before the episode ends. The perception VAE is trained in a standard self-supervised manner, with a binary cross entropy (BCE) reconstruction loss and a KL loss on the predicted distribution to enforce a normal in the encodings.

While training perception and policy *sequentially* is straightforward, we also investigate *simultaneously* training both. In the latter case, all frames observed during policy training are only stored in a buffer from which data is sampled to optimize the VAE.

**Multiple policies.** Because driving is complex to learn with a single policy, we instead consider driving as tasks ensemble and learn multiple unique policies, switched at inference. In details, we learn three independent polices for driving *straight* (SP), turning *right* (RP), turning *left* (LP). This not only helps simplifying the control module but also prevents integrating route information *in* the world model. At inference we use an external navigation system (e.g. global planning algorithm) to switch between these policies near intersections, as shown in fig. 3.

## IV. EXPERIMENTS

We conduct our experiments on Carla [7] simulator (at 10Hz) as it provides a range of high definition maps for virtual towns, and custom API calls to interact with the simulator. Navigation views maps are custom-built and rendered as 256x256 RGB images. Because these views have small domain gaps, we train only on three small sections of the track ‘Town01’ which encompass the scenarios of interest (straight and left/right turns). Evaluation is conducted on test tracks ‘Town01’ and ‘Town02’, in unseen areas, see Fig. 4.

In the following, we first provide details on the experimental setup, and evaluate the performance when perception and policy are trained simultaneously which takes as little as 8 hours on a single GPU. Finally, we ablate our contributions and compare simultaneous and sequential training.

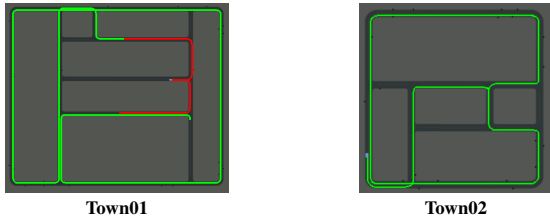


Fig. 4: **Train and test tracks layouts.** Notice we train (red) on tiny portions of Town01 road layout and test on unseen areas of Town01, Town02 (green).

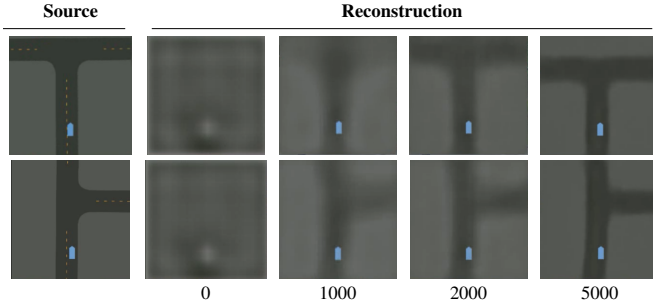


Fig. 5: **VAE reconstructions.** Qualitative reconstructions of a given source image at different training steps. The VAE quickly converges and discover main scene traits around 1000 training steps.

#### A. Experimental Setup

At each episode start, the agent are spawned with a random position on the road and orientation in  $[-45^\circ, 45^\circ]$  w.r.t. the local road curvature. A buffer limit of 50000 is used and while the buffer is being filled, the perception and control modules optimizes the world model and policy, respectively. The VAE samples from the buffer and for every iteration optimizes the world model with a batch size of 100 and epochs equivalent to the number of samples batches.

The episode duration  $T$  (seconds) is a function of complexity, in details:  $T = \min(\max(c_i, 10), 40)$ . Short episodes for low complexity prevents exploration of unwanted states and early optimization of the policy. While clamping  $T$  for larger complexity helps in optimizing the speed it also prevents unwanted wiggling, thus attaining the best policy. As optimizer we use Adam with a learning rate of  $1e-5$ , and step-wise decay with step size of 5000 and  $\gamma = 0.96$ . The trajectories collected for each episode are trained for 15 epochs with a policy and value clip of 0.1. The discount factor used is 0.99.

#### B. Performance

We evaluate our method proposal in threefold, first briefly evaluating the perception module to ensure effectiveness of our world model, second evaluating the policy performance on train tracks, and third studying the generalization capacity of our world model and policies.

**Perception.** We qualitatively evaluate the world model learned by the VAE, showing reconstructions in Fig. 5.

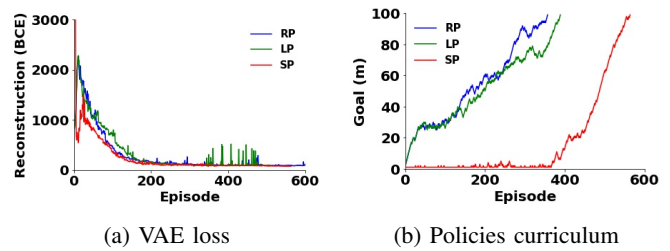


Fig. 6: **Simultaneous Perception and Policy training.** (a) VAE reconstruction loss quickly converges after only 200 episodes, while *simultaneously* the three parallel policies (b) are trained on increasing curriculum complexity.

From the latter, we note that the reconstruction improves during the training, and that only a few thousand iterations is sufficient to restore the global road layout. Quantitatively, the reconstruction loss during training in Fig. 6a also demonstrates the quick BCE loss convergence.

**Policy.** To evaluate policy, Fig. 6b shows the goal distances in meters – i.e. curriculum complexity – for all three policies trained in parallel. The turn left policy (LP) and turn right policy (RP) evolve concurrently with visible effect of our curriculum *revert* strategy since the goal partly decrease locally. Conversely, we denote that the straight policy (SP) requires significantly more episodes to optimize. We conjecture that it relates to the overcompensation problem of RL agents [17]. Ultimately, all policies converged in less than 600 episodes, with agents driving at an average speed of 10.5kmph. Since sparse reward does not enforce speed, the velocity in fact relates to the minimum acceptable speed to complete episodes being 9kmph (i.e. the episode duration  $T$  forces the agent to complete 100m in less than 40sec). We conjecture that decreasing the episode duration could further speed up the driving.

On driving style, Fig. 7 *top* shows 100 runs when driving either policy (left, straight, right) on training tracks, with markers color evolving from high speed (red) to low speed (white). On train tracks, despite the absence of any dense reward, we denote the agents successfully learned to drive in the safe drivable area. Even more, the car naturally stays close to the track center although wiggling is visible as often mentioned in the literature [57], [17]. Finally, on *Left/Right Policy* we observe that the agents learned to turn from the inside by shifting on the right or left first. While this lead to driving rules infringement we highlight that such time-optimized driving is impossible with dense rewards that penalize distance from the lane center [17].

**Generalization.** Qualitative driving on unseen test tracks, Fig. 7 *bottom*, also demonstrates the generalization of the learned policies and the world model. Despite the small domain gap of our navigation views, notice that test tracks include *unseen* road layout which both the world model and policy have to cope with. This is visible on test tracks



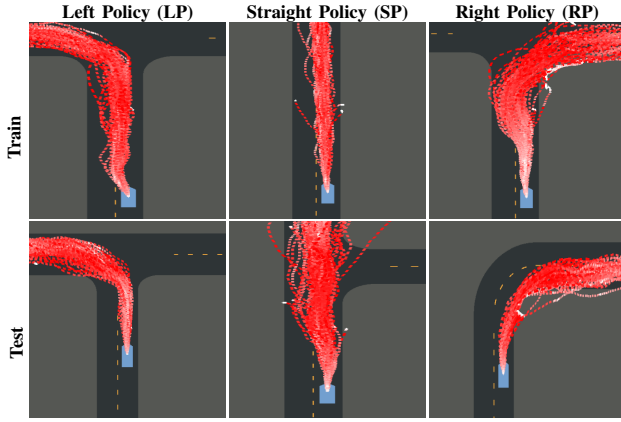


Fig. 7: **Driving style variability.** Visualisation of 100 runs for a given goal while varying the random seeds, for train tracks (Town 01) (top) or unseen test tracks (Town 01 and 02) (bottom). Our method performs similarly on either setup related to the little domain gap, and discovers proper driving behavior despite weak binary supervision. Normalized speed (white→red) show our agents learned to slow down before turns.

Method	Success rate				
	20m	50m	100m	200m	300m
<b>Ours - simultaneous</b>	<b>1.0</b>	<b>0.85</b>	<b>0.82</b>	<b>0.66</b>	<b>0.41</b>
<b>Ours - sequential</b>	0.91	<b>0.90</b>	0.90	<b>0.69</b>	<b>0.51</b>
w/o revert	0.51	0.08	0.02	0	0
w/ curr. +2	0.99	0.89	<b>0.91</b>	0.54	0.07
w/ fixed episode dur.	0.36	0.04	0	0	0
w/o constraints	0.87	0.44	0.11	0	0

TABLE I: **Driving test tracks performance.** Success rate of goal completion on *unseen test tracks* for different goal distances. The first two lines denote our pipeline with perception and policy either trained simultaneous or sequential, while bottom lines are ablations of our contributions. Even if training does not exceed 100m, policy is able to drive longer.

for example in straight policy (SP) and right policy (RP), Fig. 7 *bottom*.

To further study generalization, Tab. I reports the success rate for different goal distances on *test tracks only* (for 100 runs). From the latter, *ours simultaneous* completes 20m goal with 100% success rate, and 100m goal – the highest trained complexity – with 82% success rate. Of interest, we demonstrate that despite that agents only train with up to 100m goals, the policy scales to long driving goals like 200m (66%) and 300m (41%). Training perception and policy pipeline in a sequential manner (*ours sequential*), we denote few percent better success rates. However, this comes at the cost of more complex and twice longer training ( $\sim 18$  hours), which is why we prefer simultaneous training.

### C. Ablation

We now discuss ablation of our contributions, and report quantitative performance in Tab. I and Fig. 8.

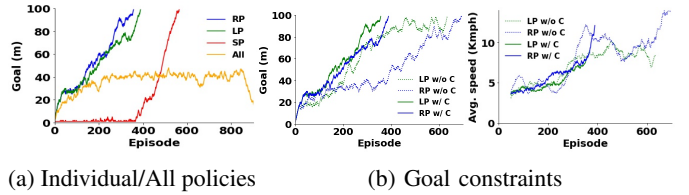


Fig. 8: (a) Effect of learning individual policies (*LP, SP, RP*) or as a single policy (*All*). (b) Comparing the performance of different policies with and without goal constraints.

**Individual policies vs Multiple policies.** To study the effect of learning multiple task simultaneously, using sparse rewards, we compare in Fig. 8a the performance of our pipeline when training either individual policies (left, straight, right) separately or all together in a single policy. From the plot, it is clearly inefficient to learn all policies (*‘All’*) in comparison to individual policies (*‘LP’*, *‘SP’*, *‘RP’*). Specifically, when learning all policies grouped, it optimizes slowly and reaches a plateau (around 40m complexity), while our individual policies all reaches max goal complexity (100m).

**Goal constraints.** We evaluate the benefit of our goal constraints described in Sec. III-B. From Fig. 8b, it is clear that without constraints (*‘w/o C’*, dotted) the policy still manages to achieve similar curriculum complexity, though speed graph shows that without constraints the agent tend to drive slower. Also, from Tab. I, without constraints the success rate is in fact significantly slower, and is even unable to generalize for long distance goals (over 100m). In the sparse reward settings, these constraints are indeed useful when an agent needs to perform a sub task along with the main goal (like driving following the traffic rules). The addition of constraints prevents the agent from exploring different actions leading to sub-optimal performance. The agent without constraints tends to deviate a lot initially in the training phase even for lesser complexity goal but learns the optimal actions with time.

**Curriculum strategies.** We also evaluate the benefit of our curriculum strategy in Tab. I. Performance without our *revert strategy* (*‘w/o revert’*) demonstrate the significant benefit of our yet simple strategy, as it avoids remaining stuck in a local minimum. Increasing complexity twice faster (*‘w/ curr +2’*) show acceptable but worse performance.

## V. CONCLUSION

We propose the first sparse reward dependent reinforcement learning agent for end to end driving. Instead of front view images, we rely on navigation maps to learn individual policies in parallel and simultaneously train perception and world model. Our results demonstrate exciting generalization capacity and natural emergence of driving styles. We also believe RL driving with sparse rewards open doors to new perspective such as the inclusion of driving rules – a major RL challenge.

## REFERENCES

- [1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *ICML*, 2009.
- [2] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. W. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang. Dota 2 with large scale deep reinforcement learning. *ArXiv*, 2019.
- [3] J. Chen, B. Yuan, and M. Tomizuka. Model-free deep reinforcement learning for urban autonomous driving. *ITSC*, 2019.
- [4] J.-Y. Chen, S. E. Li, and M. Tomizuka. Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. *ArXiv*, 2020.
- [5] V. Dasagi, J. Bruce, T. Peynot, and J. Leitner. Ctrl-z: Recovering from instability in reinforcement learning. *ArXiv*, 2019.
- [6] D. Dolgov. Practical search techniques in path planning for autonomous driving. 2008.
- [7] A. Dosovitskiy, G. Ros, F. Codevilla, A. López, and V. Koltun. Carla: An open urban driving simulator. In *CoRL*, 2017.
- [8] J. Elman. Learning and development in neural networks: the importance of starting small. *Cognition*, 1993.
- [9] A. Graves, M. G. Bellemare, J. Menick, R. Munos, and K. Kavukcuoglu. Automated curriculum learning for neural networks. In *ICML*, 2017.
- [10] S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. *ICRA*, 2017.
- [11] D. R. Ha and J. Schmidhuber. World models. *NeurIPS*, 2018.
- [12] J. Hare. Dealing with sparse rewards in reinforcement learning. *ArXiv*, 2019.
- [13] S. Hecker, D. Dai, and L. Van Gool. Learning accurate, comfortable and human-like driving. *arXiv*, 2019.
- [14] D. Held, X. Geng, C. Florensa, and P. Abbeel. Automatic goal generation for reinforcement learning agents. *PMLR*, 2018.
- [15] C. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, and M. J. Kochenderfer. Combining planning and deep reinforcement learning in tactical decision making for autonomous driving. *T-IV*, 2020.
- [16] Y. Hu, W. Wang, H. Jia, Y. Wang, Y. Chen, J. Hao, F. Wu, and C. Fan. Learning to utilize shaping rewards: A new approach of reward shaping. In *NeurIPS*, 2020.
- [17] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot, and F. Nashashibi. End-to-end race driving with deep reinforcement learning. *ICRA*, 2018.
- [18] N. Jiang, S. Jin, and C. Zhang. Hierarchical automatic curriculum learning: Converting a sparse reward navigation task into dense reward. *Neurocomputing*, 2019.
- [19] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *ArXiv*, 2018.
- [20] C. Katrakazas, M. Quddus, W. hua Chen, and L. Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C-emerging Technologies*, 2015.
- [21] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah. Learning to drive in a day. *ICRA*, 2019.
- [22] F. Khan, X. Zhu, and B. Mutlu. How do humans teach: On curriculum learning and teaching dimension. In *NeurIPS*, 2011.
- [23] A. Laud and G. DeJong. The influence of reward on the speed of reinforcement learning: An analysis of shaping. In *ICML*, 2003.
- [24] A. Lazaric, M. Restelli, and A. Bonarini. Transfer of samples in batch reinforcement learning. In *ICML*, 2008.
- [25] C. Li and K. Czarnecki. Urban driving with multi-objective deep reinforcement learning. In *AAMAS*, 2019.
- [26] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *ICLR*, 2016.
- [27] Y. Luo, K. Dong, L. Zhao, Z. Sun, C. Zhou, and B. Song. Balance between efficient and effective learning: Dense2sparse reward shaping for robot manipulation with environment uncertainty. *CoRR*, 2020.
- [28] O. Marom and B. Rosman. Belief reward shaping in reinforcement learning. In *AAAI*, 2018.
- [29] T. Matisen, A. Oliver, T. Cohen, and J. Schulman. Teacher-student curriculum learning. *Transactions on Neural Networks and Learning Systems*, 2020.
- [30] M. Meghjani, Y.-F. Luo, Q. H. Ho, P. Cai, S. Verma, D. Rus, and D. D. Hsu. Context and intention aware planning for urban driving. *IROS*, 2019.
- [31] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016.
- [32] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing atari with deep reinforcement learning. *ArXiv*, 2013.
- [33] P. K. Mohanty and D. Parhi. Cuckoo search algorithm for the mobile robot navigation. In *SEMCCO*, 2013.
- [34] S. Nagesh Rao, H. E. Tseng, and D. Filev. Autonomous highway driving using deep reinforcement learning. *SMC*, 2019.
- [35] K. Napat, M. I. Valls, D. Hoeller, and M. Hutter. Practical reinforcement learning for mpc: Learning from sparse objectives in under an hour on a real robot. In *L4DC 2020*, 2020.
- [36] A. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, 1999.
- [37] X. Pan, Y. You, Z. Wang, and C. Lu. Virtual to real reinforcement learning for autonomous driving. In *BMVC*, 2017.
- [38] X. Pan, Y. You, Z. Wang, and C. Lu. Virtual to real reinforcement learning for autonomous driving. 2017.
- [39] R. Portelas, C. Colas, K. Hofmann, and P.-Y. Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *CoRL*, 2019.
- [40] S. Racanière, A. Lampinen, A. Santoro, D. P. Reichert, V. Firoiu, and T. Lillicrap. Automated curricula through setter-solver interactions. *ArXiv*, 2019.
- [41] J. Randlev and P. Alström. Learning to drive a bicycle using reinforcement learning and shaping. In *ICML*, 1998.
- [42] M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degraeve, T. Wiele, V. Mnih, N. Heess, and J. T. Springenberg. Learning by playing solving sparse reward tasks from scratch. In *ICML*, 2018.
- [43] D. L. T. Rohde and D. Plaut. Language acquisition in the absence of explicit negative evidence: how important is starting small? *Cognition*, 1999.
- [44] T. Sanger. Neural network learning control of robot manipulators using gradually increasing task difficulty. *T-RO*, 1994.
- [45] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, 2017.
- [46] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.
- [47] S. Sukhbaatar, Z. Lin, I. Kostrikov, G. Synnaeve, A. Szlam, and R. Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. In *ICLR*, 2018.
- [48] A. Tampuu, T. Matisen, M. Semikin, D. Fishman, and N. Muhammad. A survey of end-to-end driving: Architectures and training methods. *Transactions on Neural Networks and Learning Systems*, 2020.
- [49] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *J. Mach. Learn. Res.*, 2009.
- [50] A. C. Tenorio-González, E. Morales, and L. Pineda. Dynamic reward shaping: Training a robot by voice. In *IBERAMIA*, 2010.
- [51] M. Toromanoff, E. Wirbel, and F. Moutarde. Deep reinforcement learning for autonomous driving. In *CVPR*, 2019.
- [52] M. Toromanoff, E. Wirbel, and F. Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *CVPR*, 2020.
- [53] M. Toromanoff, E. Wirbel, F. Wilhelm, C. Vejarano, X. Perrotton, and F. Moutarde. End to end vehicle lateral control using a single fisheye camera. In *IROS*, 2018.
- [54] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv*, 2017.
- [55] C. Wang, J. Wang, J. Wang, and X. Zhang. Deep-reinforcement-learning-based autonomous uav navigation with sparse rewards. *Internet of Things Journal*, 2020.
- [56] D. Weinshall, G. Cohen, and D. Amir. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *ICML*, 2018.
- [57] P. Wolf, C. Hubschneider, M. Weber, A. Bauer, J. Härtl, F. Dürr, and J. M. Zöllner. Learning how to drive in a real world simulation with deep q-networks. In *IV*, 2017.