HAL open science

GRI: General Reinforced Imitation and its Application to Vision-Based Autonomous Driving

Raphael Chekroun, Marin Toromanoff, Sascha Hornauer, Fabien Moutarde

To cite this version:

Raphael Chekroun, Marin Toromanoff, Sascha Hornauer, Fabien Moutarde. GRI: General Reinforced Imitation and its Application to Vision-Based Autonomous Driving. NeurIPS 2021, Machine Learning for Autonomous Driving Workshop, Dec 2021, Virtual, United States. hal-03442328v3

HAL Id: hal-03442328

https://hal.science/hal-03442328v3

Submitted on 30 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

GRI: General Reinforced Imitation and its Application to Vision-Based Autonomous Driving

Raphael Chekroun[1,2], Marin Toromanoff[2], Sascha Hornauer[1], Fabien Moutarde[1]
[1]Mines Paris - PSL, [2]Valeo DAR
firstname.lastname@{minesparis.psl.eu, valeo.com}

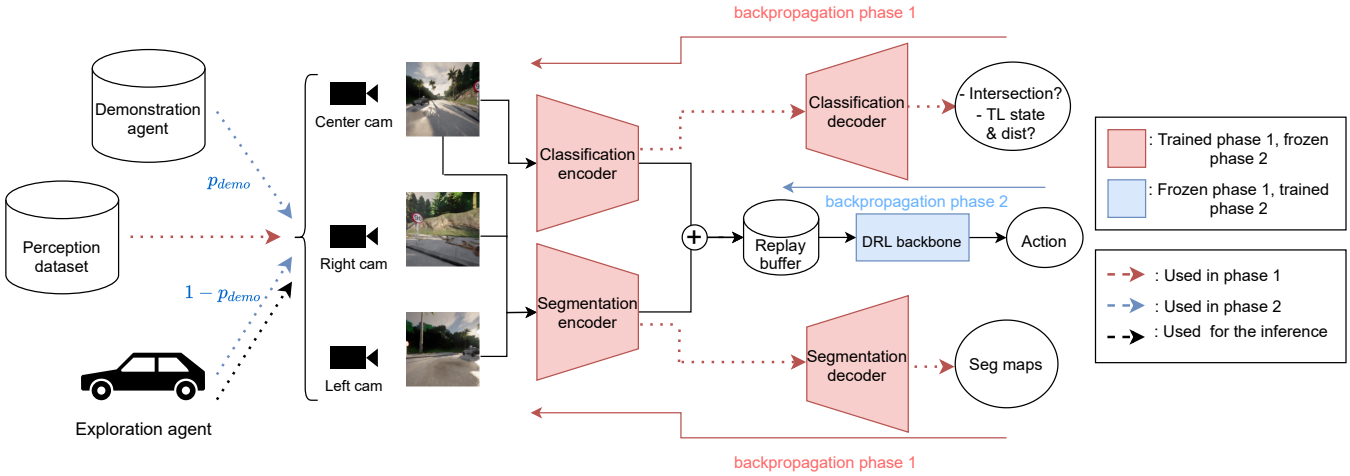

Fig. 1: General Reinforced Imitation (GRI) is applied to visual-based autonomous driving in an end-to-end pipeline composed of a perception module encoding RGB images from the three cameras on the driving agent and a decision-making module inferring an action from the encoded features. This pipeline is trained in two phases: (1) Visual encoders are pretrained on a perception dataset on several auxiliary tasks, which are semantic segmentation, road type classification, relevant traffic light presence, and if there is such a traffic light, its state and the distance to it. (2) Visual encoders are frozen and a GRI-based DRL network is trained with both pre-generated expert data with an offline demonstration agent and an online exploration agent gathering data from a simulator. At any given training step, the next episode to add to the replay buffer comes from the demonstration agent with a probability of p_{demo}, else from the exploration agent. Actions correspond to a pair (steering, throttle) to apply to the car.

***Abstract*— Deep reinforcement learning (DRL) has been demonstrated to be effective for several complex decision-making applications such as autonomous driving and robotics. However, DRL is notoriously limited by its high sample complexity and its lack of stability. Prior knowledge, e.g. as expert demonstrations, is often available but challenging to leverage to mitigate these issues. In this paper, we propose General Reinforced Imitation (GRI), a novel method which combines benefits from exploration and expert data and is straightforward to implement over any off-policy RL algorithm. We make one simplifying hypothesis: expert demonstrations can be seen as perfect data whose underlying policy gets a constant high reward. Based on this assumption, GRI introduces the notion of offline demonstration agents. This agent sends expert data which are processed both concurrently and indistinguishably with the experiences coming from the online RL exploration agent. We show that our approach enables major improvements on camera-based autonomous driving in urban environments. We further validate the GRI method on Mujoco continuous control tasks with different off-policy RL algorithms. Our method ranked first on the CARLA Leaderboard and outperforms World on Rails, the previous state-of-the-art, by 17%.**

I. INTRODUCTION

Autonomous driving (AD) in urban areas is a convoluted task. Agents have to efficiently analyse a highly complex environment and make online decisions to follow driving rules whilst simultaneously interacting with other dynamic agents, such as drivers or pedestrians. That is why, literature in autonomous driving focuses on different learning methods rather than the design of general hand-crafted rules.

Imitation learning (IL) [1], [2], [3], [4], especially behavior cloning, aims at mimicking expert behavior for a given task. It requires a significant amount of annotated data, often recorded by human drivers. Even though this kind of data can be recorded easily on a large scale, practical safety concerns in real traffic lead to heavily biased observations showing predominantly safe driving examples, and under-represents rare dangerous situations. Hence, IL agents suffer from distribution mismatch and will struggle to recover from its own mistakes.

Deep reinforcement learning (DRL) [5], [6], [7], [8] offers an alternative, more robust to distribution mismatch than IL, by letting the agent learn from its own mistakes through trial-and-error. In the RL framework, the agent explores its

environment by itself and gathers rewards, a numerical value assessing how much a given action in a given state is good. The goal of the agent is to maximize its cumulative rewards. To do so, the agent needs to optimize sequences of actions rather than instantaneous ones. Nonetheless, DRL needs an order of magnitude more data than IL to converge due to this extensive, and often time-consuming, exploration of the environment during the training.

To overcome IL distribution mismatch and RL data inefficiency, we propose General Reinforced Imitation (GRI), a novel method which combines both exploration and prior knowledge from demonstrations. GRI is based on the simplifying hypothesis that expert data presents a perfect behavior and therefore, an expert's action should receive a constant, high reward. Straightforward to implement over any off-policy algorithm, GRI introduces the notion of an offline *demonstration agent*. This offline agent sends expert data associated with a constant demonstration reward to the replay buffer of an RL online exploration agent. We note that those expert data are processed by the DRL algorithm concurrently and indistinguishably from the exploration data

The GRI method is applied to visual-based autonomous driving in an end-to-end pipeline on the CARLA simulator [9], an open-source simulator for research in autonomous driving. We call this algorithm GRI for Autonomous Driving (GRIAD). The whole pipeline is represented in Fig. 1. On the CARLA Leaderboard, an online benchmark ranking agents according to the quality of their driving, we achieved 17% better results than World on Rails [10], the prior top ranking entry. In addition, our method used only three cameras and no LiDAR, which is fewer sensors than the other top entries [10], [3]. At the time of writing (February 2022), GRIAD is the best camera-based agent on the CARLA Leaderboard according to the main metric, the driving score. We also conducted ablation studies to highlight the impact of GRIAD compared to standard RL training on the CARLA NoCrash Benchmark [11].

Finally, we conducted experiments on the Mujoco [12] benchmark to investigate our method adaptability and generalizability. Tests were conducted on four different Mujoco environments, with two different DRL algorithm as backbones. Our experiments demonstrate that using the GRI framework systematically leads to better results, even when the expert data are noisy or not significantly better than the trained vanilla RL algorithm.

We summarize our main contributions below:

- Definition of the novel GRI method to combine offline demonstrations and online exploration.
- Presentation and ablation study of GRI for visual-based Autonomous Driving (GRIAD) algorithm.
- Further analysis of GRI-based algorithms on the Mujoco benchmark.

II. RELATED WORK

The GRI method aims at leveraging both offline expert demonstrations and online simulator exploration. Our main application is end-to-end camera-based autonomous driving on the CARLA simulator [9]. Therefore, this section focuses both on end-to-end autonomous driving methods that achieved milestones on CARLA, and existing decision-making methods learning from demonstration and exploration.

A. End-to-end Autonomous Driving on CARLA

End-to-end autonomous driving, i.e. directly mapping sensor signals to control is a highly complex task on which training an agent with DRL is tedious. IL methods were the first to lead the CARLA Leaderboard. In particular, Learning by Cheating (LBC) [13] presents an efficient method to train a behavior cloning agent in two steps: (i) train a privileged behavior cloning agent which has access to all the ground truth data, and (ii) train a behavior cloning agent to mimic the privileged one. An evaluation of several methods on the NoCrash benchmark, presented in Chen et al. [10], shows that LBC presents great results on the training conditions but generalizes poorly on unknown environments.

DRL can also be used for end-to-end autonomous driving. However, camera-based DRL comes with some drawbacks. Indeed, image inputs are often of high dimensions thus requiring larger DRL networks which are usually difficult to train to convergence. Therefore, for camera-based DRL, one can encode the sensors' signal in a more compact and semantically rich representation to train the DRL network on this predefined latent space as in D. Gordon et al. [14]. This latent space can be obtained by pretraining a visual encoder on some visual tasks, such as segmentation or classification.

Based on this principle, Toromanoff et al. [15] introduced the *Implicit Affordances* (IAs) method. They design and train an efficient DRL agent on CARLA, winning the CARLA challenge two years in a row. To do so, they propose an end-to-end pipeline composed of two subsystems trained successively. First, a visual encoder is trained on some auxiliary tasks. Those tasks are semantic segmentation, classification of the type of road, detection of traffic lights, and if there is a relevant traffic light, the state of, and distance to, the light. Then, the visual encoder is frozen and the DRL-based decision-making subsystem is trained on the encoder latent space.

Another top ranked camera-based agent on the CARLA Leaderboard is World on Rails [10] which assumes the world to be on rails, meaning that the agent's actions affect only its own state and do not influence its environment. Based on that hypothesis, they transpose the driving problem into a simple, yet powerful, tabular RL setup.

Finally, Transfuser [3] and Learning from All Vehicles, more recent top ranked agents on the CARLA Leaderboard, mainly focuses on LiDAR and camera fusion.

Other concurrent work combines an RL driving coach and an IL learner, mediated with a learned bird's map [16] but are not currently in the CARLA Leaderboard.

B. Learning from demonstration and exploration

The aforementioned IL and RL strengths and weaknesses are complementary. Indeed, IL suffers from distribution

mismatch contrarily to online RL. Alternatively, as RL learns from scratch it is less data efficient than IL which incorporates prior demonstration knowledge during training.

To take the best of both worlds, some algorithms combine IL and RL to maximize efficacy by leveraging both expert data and exploration [17], [18], [19], [20]. In particular, demonstrations can be used to initialize policies by pre-training the network [21], [19], [17] or leveraged with a specifically designed reward [17], [18].

Soft-Q Imitation Learning (SQIL) [18] and Deep Q-learning from Demonstrations (DQfD) [17] are the two closest approaches to ours as both take advantage of demonstrations in a different way and can be applied to any off-policy RL algorithms.

SQIL [18] does IL using an RL agent. To do so, the replay buffer is initially filled with demonstrations, associated with a constant reward $r_{demo} = 1$. A RL agent collects data from exploration into the replay buffer, associated with a constant reward $r_{explo} = 0$. Thus, SQIL designed an RL agent that learns to imitate expert behavior, and has been mathematically demonstrated to be equivalent to regularized behavior cloning. However, SQIL does not efficiently leverage exploration as environment rewards are never used. Our method combines both the IL part from SQIL and the classical, rich RL online exploration.

DQfD [17] is based on DQN [5], an off-policy RL algorithm with a replay buffer. DQfD first pretrains the agent on expert data with both IL and RL losses using the real reward given by the environment. After some steps of pretraining, the agent starts gathering data from the environment in the memory buffer. The network is then trained on batches composed of exploration data with a RL loss and expert data with both IL and RL losses. Nonetheless, DQfD uses simultaneously reinforcement and imitation which can have divergent losses and are difficult to jointly optimize [22]. Our method leverages demonstrations and exploration exclusively with an RL loss, and thus cannot suffer from divergent losses issue. Moreover, DQfD, contrarily to GRI, relies on the true environment reward for the expert data, which cannot always be obtained.

III. GENERAL REINFORCED IMITATION (GRI)

Our pipeline for autonomous driving is an end-to-end system. Its decision-making subsystem uses the GRIAD algorithm, an adaptation of the GRI method to visual-based autonomous driving (AD) on CARLA. This section presents the GRI method and details the whole pipeline.

A. Method

GRI is a method which is straightforward to implement over any off-policy RL algorithm using a replay buffer, such as SAC [23], DDPG [24], DQN [5] and its successive improvements [25], [26]. GRI is built upon the hypothesis that expert demonstrations can be seen as perfect data whose underlying policy gets a constant high reward. We denote this as demonstration reward, r_{demo}. In our experiments we chose r_{demo} to be the maximum of the reward.

Algorithm 1: GRI: General Reinforced Imitation

```
Input: r_demo demonstration reward value, p_demo
  probability to use demonstration agent;
Initialize empty buffer 𝒷;
while not converged do
    if len(𝒷) ≥ min_buffer then
        do a DRL network update;
    end
    if random.random() ≥ p_demo then
        collect episode (s_t^online, a_t, r_t, s_{t+1}^online)_t in
          buffer 𝒷 with exploration agent
    else
        add episode (s_t^offline, a_t, r_demo, s_{t+1}^offline)_t in
          buffer 𝒷 with demonstration agent;
    end
end
```

The idea of GRI is to distill expert knowledge from demonstrations into an RL agent during the training phase. To do so, we defined two types of agents: (i) the online *exploration agent*, which is the regular RL agent exploring its environment to gather experiences $(s_t^{online}, a_t, r_t, s_{t+1}^{online})$ into the memory buffer, and (ii) the offline *demonstration agent* which sends expert data associated with a constant demonstration reward $(s_t^{offline}, a_t, r_{demo}, s_{t+1}^{offline})$ to the memory buffer. s_t is the state, a_t the chosen action and r_t the reward at time t. At any given training step, the next episode to add to the replay buffer comes from the demonstration agent with a probability of p_{demo}, else from the exploration agent. GRI is summarized in Algorithm 1.

B. GRI for Autonomous Driving

We applied GRI on a pipeline inspired by Toromanoff et al. Implicit Affordances method [15]. As this method is trained in two phases, hence making it modular, we optimized both the visual and the decision-making subsystems independently.

Design of the vision subsystem We first train two visual encoders on segmentation and classifications tasks with different camera perspectives to extract compact semantic features.

We found that single camera setup is leading to more collisions on intersections, as sensors are not able to see close obstacles while turning. Thus, we mounted three RGB cameras on the hood of our agent vehicle, at the coordinates $x = 2.5m, z = 1.2m$ and $y \in \{-0.8, 0, 0.8\}m$ relatively to the center of the car. The side cameras are angled at 70°. All three cameras have a 100° field of view.

Our visual subsystem is composed of two highly specialized Efficientnet-b1 [27] models, one for the segmentation and one for the classification and regression tasks, as shown in Fig. 2. We concatenate the four outputs (three segmentations, one for each camera, and one classification from the

front camera only) of both Efficientnet-b1 and use it similarly as Implicit Affordances (IAs) for the DRL training.

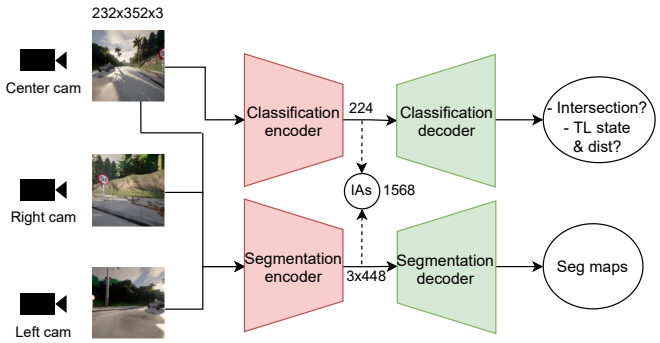

Fig. 2: Feature extraction from RGB camera images for the visual subsystem. Two encoder-decoder networks are pretrained on segmentation, classifications and regression tasks. Classifications and regression are only performed on the center image while all three images are segmented. After training, the visual encoders serve as fixed feature extractors with frozen weights. For the DRL backbone training, both encoder outputs are concatenated and sent to the memory buffer as input to DRL. Both encoders are Efficientnet-b1. The segmentation decoder is fully convolutional, and the classification decoder is an MLP with several outputs.

This architecture allows to keep the same accuracy on classification and segmentation metrics as if we were using a single bigger encoder for all the auxiliary tasks like in Toromanoff et al. [15], while reducing the encoder latent space dimension by a factor of ~ 5.

The visual part for the CARLA Leaderboard has been trained on a dataset of 400,000 samples which corresponds to 44 hours of driving. This dataset has been generated with the CARLA autopilot on every town with random trajectories. Each sample of the dataset is composed of three images from the three cameras and the corresponding ground truth information, which are segmentation maps from CARLA, booleans indicating the presence of an intersection and the presence of a traffic light in front of the car. And if there is a traffic light, a class corresponding to its color, and the distance to it in meters. Trajectories have been augmented with random cameras translations and rotations.

Design of the Decision Subsystem The decision subsystem takes as input four consecutive encodings of the three camera images and outputs an action. Therefore, a state contains visual features from the last 300 milliseconds, as the simulator runs at 10 FPS. An action is defined by the combination of the desired steering of the wheel, and the throttle or brake to apply.

Generating data on the CARLA simulator is very computationally expensive. We used a Rainbow-IQN Ape-X [28], which is a distributed DRL backbone, to mitigate this issue.

Due to Rainbow-IQN Ape-X being based on DQN [5] the action state has to be discrete. Therefore, we discretized the action state in 27 steering values, and 4 throttle or brake values. The discretized action space contains $27 \times 4 = 108$ actions.

We called this setup GRI for Autonomous Driving (GRIAD). We diagram it in Fig. 3.

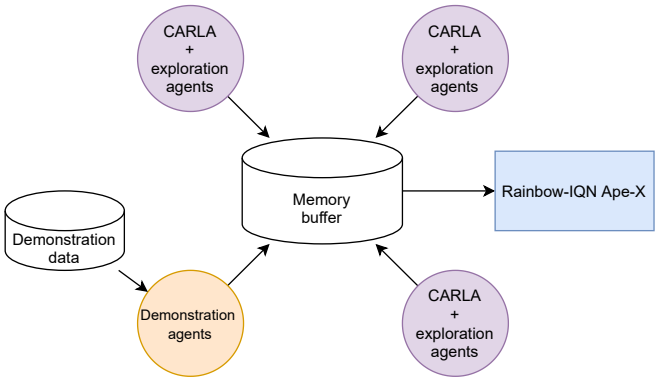

Fig. 3: Simplified representation of the distributed GRIAD setup with a Rainbow-IQN Ape-X backbone. A central computer receives data in a shared replay buffer from both exploration and demonstration agents running on other computers. Data are sampled from this replay buffer to make the backpropagation and update the weights of all the agents. Images from the agents are encoded using the network presented in Fig. 2 before being stored in the memory buffer.

The demonstration dataset contains 200,000 samples, which correspond to 22 hours of driving, generated using the autopilot from CARLA on predefined tracks published by the CARLA team[1]. Each sample from the demonstration dataset consists of three images from the three cameras and a discrete action obtained by mapping continuous actions of the expert to our discrete set of RL actions. We did not use any data augmentation. We note that the autopilot makes driving errors such as collisions, red light infractions, or the car getting stuck for hundreds of frames. As a result $\sim 10\%$ of our demonstrations correspond to poor action choices. However, we decided to use this demonstration dataset as it is in order to assess the robustness of our method to noisy demonstrations.

In our experiments on CARLA, GRIAD had a total of 12 agents, including 3 demonstration agents, running in a distributed setup and sending data to the memory buffer. As demonstration agents have been constrained to send data at the same frequency as exploration agents, this is equivalent to having $p_{demo} = 25\%$.

The reward function used for the exploration agents is the same as in Toromanoff et al. [15]. Since this reward has a range between 0 and 1, we set the demonstration reward to $r_{demo} = 1$.

IV. EXPERIMENTAL RESULTS

The GRI method was assessed on its primary application of visual-based autonomous driving on the CARLA Leaderboard and with an ablation study comparing it to vanilla RL. Further studies of the method have also been conducted on the Mujoco benchmark to analyze its behavior depending on the proportion of demonstration agents and highlight its generalizability to other DRL backbones.

[1] https://github.com/carla-simulator/scenario_runner

A. GRIAD on CARLA

On the CARLA leaderboard. We trained GRIAD for 60M steps (∼45M exploration steps + 200,000 expert data sampled ∼15M times). Both visual and decision-making parts were trained on all available maps with all available weather. We compare to LBC [13], IAs [15], Transfuser+ [3] and World on Rails [10]. Our method outperforms World on Rails, the previous leading method on the CARLA leaderboard, by $\sim 17\%$ on the main metric, the driving score, while using fewer sensors. However, more recent LiDAR-based methods give significantly better results, but cannot be compared directly as inputs are different. CARLA Leaderboard results are presented in Table I.

Method	Cam.	LiDAR	DS	RC	IS
GRIAD (ours)	3	✗	**36.79**	**61.85**	**0.60**
Rails [10]	4	✗	31.37	57.65	0.56
IAs [15]	1	✗	24.98	46.97	0.52
LBC [13]	3	✗	10.9	21.3	0.55
LAV	4	✓	**61.8**	**94.5**	**0.64**
Transfuser+ [3]	4	✓	50.5	73.8	0.68

TABLE I: Comparison of camera-based and LiDAR-based agents on CARLA Leaderboard's driving metrics: driving score (DS, main metric), route completion (RC), and infraction score (IS). Results from the public CARLA Leaderboard website on February 2022. Higher is better for all metrics. Our method improves the driving score by 17% relative to the prior camera-based state-of-the-art [10] while using fewer sensors than the two other best.

Ablation study on the NoCrash benchmark. We compared GRIAD to regular RL by using the same architecture with and without demonstration agents on the NoCrash benchmark [11]. To do so, agents are trained on a single environment (Town01) under a specific set of training weather. Then, agents are evaluated on several scenarios with different traffic density on the training (Town01) and test (Town02) town with training and test sets of weather.

Task	Town, Weather	RL 12M	RL 16M	GRIAD
Empty	train, train	96.3 ± 1.5	**98.0 ± 1.0**	**98.0 ± 1.7**
Regular		95.0 ± 2.4	**98.6 ± 1.2**	**98.3 ± 1.7**
Dense		91.7 ± 2.0	**95.0 ± 1.6**	93.7 ± 1.7
Empty	test, train	83.3 ± 3.7	**96.3 ± 1.7**	94.0 ± 1.6
Regular		82.6 ± 3.7	**96.3 ± 2.5**	93.0 ± 0.8
Dense		61.6 ± 2.0	**78.0 ± 2.8**	**77.7 ± 4.5**
Empty	train, test	67.3 ± 1.9	73.3 ± 2.5	**83.3 ± 2.5**
Regular		76.7 ± 2.5	81.3 ± 2.5	**86.7 ± 2.5**
Dense		67.3 ± 2.5	80.0 ± 1.6	**82.6 ± 0.9**
Empty	test, test	60.6 ± 2.5	62.0 ± 1.6	**68.7 ± 0.9**
Regular		59.3 ± 2.5	56.7 ± 3.4	**63.3 ± 2.5**
Dense		40.0 ± 1.6	46.0 ± 3.3	**52.0 ± 4.3**

TABLE II: Ablation study of GRIAD using the NoCrash benchmark. Mean and standard deviation over 3 evaluation seeds. Score is the percentage of road completed without any crash. GRIAD experimentally shows to generalize more on test weather than RL with 12M and 16M steps and globally gives the best agent. We note that, for computational reason, neither the RL nor GRIAD was trained until convergence. Hence, comparison to the state-of-the-art should rather be done on the CARLA Leaderboard with the fully trained GRIAD model, cf Table I.

For these experiments, GRIAD was trained on 16M samples corresponding to 12M exploration steps + 25,000 expert data which have been sampled 4M times in total. We present an ablation study to show how GRIAD compares to RL without GRI i.e. without demonstration agents, using two vanilla RL models: one trained on 12M exploration steps and the other on 16M exploration steps. Each agent was trained using the exact same visual encoder trained on another demonstration dataset of 100,000 samples coming exclusively from Town01 under training weather. Results are presented in Table II.

We first observe that GRIAD systematically gives better results than RL with 12M steps, while taking approximately the same time to train (+$\sim 4\%$). Indeed, as demonstration agents do not require any interaction with the simulator, we can add them at a negligible cost and still improve results.

We also observe that while RL with 16M steps does better than GRIAD on train weather, GRIAD gives better results on the test weather while being $\sim 25\%$ faster to train. We believe this is because RL tends to overfit on a given environment if it explores it too much. Hence, replacing 4M exploration data with 25,000 demonstration data sampled ~ 160 times each appears to reduce the overfitting and allows a better generalization.

We also trained the same pipeline using the SQIL method during 20M steps, but the evaluation reward stayed particularly low during training. First test showed SQIL to be inefficient for autonomous driving on CARLA as it did not learn to drive at all, staying static or drifting off the road most of the time. It reached the score of 0 on every evaluated tasks. We believe that the reward signal as defined by SQIL is not rich enough to allow the network to converge on such a highly complex task.

B. GRI on the Mujoco benchmark

To further validate the GRI method we conducted experiments on selected Mujoco [12] environments, shown in Fig. 4. Expert data were generated using chainerrl [29] pretrained RL agent weights and contain 200,000 samples. For each environment, the value of r_{demo} was chosen as the highest value chainerrl expert agent reached during the generation of the dataset. As we did not find real expert data on Mujoco environments, expert data is not always significantly better than our trained vanilla RL network. Hence, this study assesses the efficiency of GRI even with suboptimal expert data.

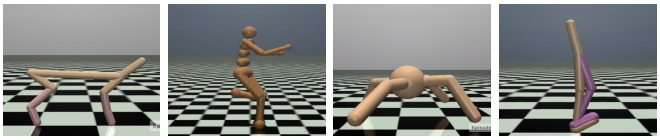

Fig. 4: Mujoco environments used for our experiments. Respectively HalfCheetah-v2, Humanoid-v2, Ant-v2, and Walker2d-v2. Articulations are controlled to make them walk. Rewards depends on the covered distance.

Study on the proportion of demonstration agents. Experiments being faster on Mujoco environments than on CARLA we were able to investigate the impact of the proportion of demonstration agents. For these experiments we used a GRI-SAC i.e. a GRI algorithm using SAC [23] as DRL backbone, and we vary the proportion of demonstration

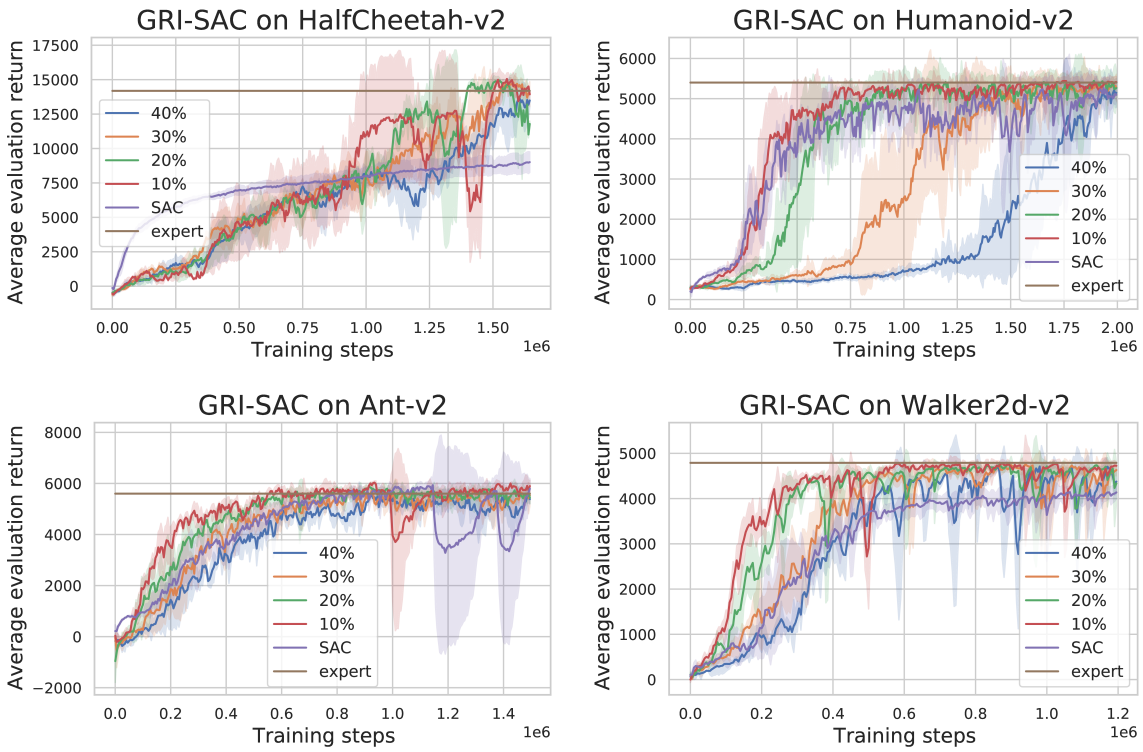

Fig. 5: Evolution of the evaluation reward on Mujoco environments with different proportions of demonstration agents with GRI-SAC. GRI-SAC with 0% demonstration agent is vanilla SAC. We observe that GRI-SAC always reaches the level of the expert even when the expert is significantly better than the trained vanilla SAC. The proportion of demonstration agent have a significant impact on the dynamic of the convergence.

agents between 0% and 40%. Each experiment has been repeated three times, with different seeds. Fig. 5 presents the results with the variances and the evaluation reward of the expert. Experiments were conducted with public code from GitHub[2] which has been adapted with GRI.

We observe three different dynamics.

- For HalfCheetah-v2, a difficult task on which the expert is significantly stronger than the trained SAC, we observe that the beginning of the training is slower using GRI-SAC; we call this a *warm up phase* which we will explain further in Section IV-C. However the rewards turns out to become significantly higher after some time. On this game, GRI-SAC is better than SAC with every proportion of demonstration agents. Best scores were reached with 10% and 20% of demonstration agents.
- For Humanoid-v2, a difficult task on which the expert is just a little stronger than the trained SAC, we observe that the higher the number of demonstration agents is, the longer the *warm up phase* is. Nonetheless, GRI-SAC models end up having higher rewards after their warm up phase. Best scores are reached with 10% and 20% of demonstration agents.
- Ant-v2 and Walker2d-v2 are the easiest tasks of the four evaluated. On Ant-v2 the SAC agent reaches the expert level, converging similarly as GRI-SAC regardless of the number of demonstration agents used. Nevertheless, GRI-SAC converges faster with 10% and 20% demonstration agents. On Walker2d-v2 the final reward of GRI-SAC is significantly higher and reaches the expert level, while SAC remains below.

More experiments were conducted with the proportion of demonstration agents varying between 50% and 90%. Results were significantly worse than using 20% demonstration agents. We therefore conclude that the proportion of demonstration agent should not exceed 50%. We discuss some qualitative insights in the limitation sections.

These experiments reveals, at least on the evaluated Mujoco environments, that 20% demonstration agents seems to be the best choice for GRI-SAC to reach the expert level.

1) GRI with DDPG as DRL backbone: We also investigated the contribution of the DRL backbone to assess the generalizability of the GRI method. To do so, we evaluated the same tasks with the Deep Deterministic Policy Gradient (DDPG) algorithm [24] instead of SAC. For these experiments, we fixed the proportion of demonstration agents to 20%. Results are shown in Fig. 6.

We observe that, alike to GRI-SAC with a proportion of 20% demonstration agents, GRI-DDPG reaches better results than DDPG on all the tested environments. However, GRI-DDPG does not systematically reach the level of the expert. While final rewards are better with SAC and GRI-SAC, the dynamics of the rewards evolution is about the same with both backbones, cf. Fig. 5. We can conclude that GRI is easily adaptable and generalizes to locomotion tasks where

[2]Original code from `https://github.com/dongminlee94/deep_rl`

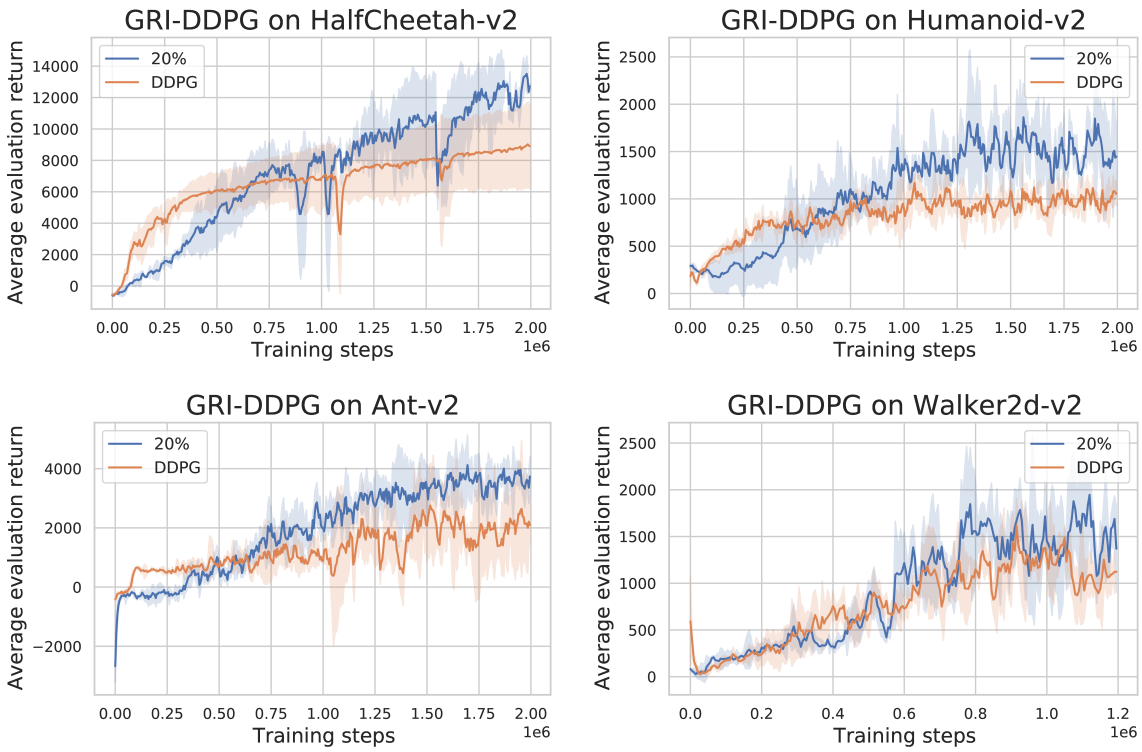

Fig. 6: Comparaison of the evaluation reward evolution on Mujoco environments between GRI-DDPG with 20% of demonstration agents and vanilla DDPG. GRI-DDPG systematically leads to a better reward than vanilla DDPG. However, contrarily to GRI-SAC, GRI-DDPG with 20% demonstration agents does not systematically reach the expert level.

it robustly outperforms two alternative methods.

C. Limitations

The main limitations of this method are consequences of our initial hypothesis that demonstration data can always be associated with a constant maximal reward r_{demo}.

A first limitation occurs if the demonstration data is not constantly optimal, e.g. due to low expert performance on some aspect of a given task, as this introduces noise in the reward function. This is the case in our demonstration dataset on the CARLA simulator, as expert data have been generated using an imperfect autopilot containing $\sim 10\%$ noisy demonstrations. Still, GRIAD showed to improve our model by a significant margin over vanilla RL. Therefore, we can consider the GRI setup to present some robustness to noisy demonstrations.

A second limitation of our approach is the warm-up phase on some difficult environments, as observed in Fig. 5 on HalfCheetah-v2 and Humanoid-v2. This warm up phase can be seen as the consequence of a distribution shift. Indeed, GRI suffers from a sort of distribution shift when the training expert data mostly represent actions made in states not reached yet by the exploration agents. In particular, we observed this effect on HalfCheetah-v2: the expert agent does not walk but jumps as soon as it touches the ground, which is a complex yet highly efficient strategy. But to reach a state where it can successfully jump, it needs some warm up to gain the required speed and momentum by doing some low reward actions. Hence, our GRI-SAC agents learns to jump before it is able to walk, making it fall. Once the agent learned how to reach the jumping state, reward steadily increase until convergence. However, we observe that the lower the proportion of demonstration agents is, the faster the model is able to recover from this distribution shift. Indeed, collecting more exploration data following the current agent policy compensates the distribution shift between demonstration and exploration data.

Finally, a third limitation of our approach is the inconsistency of the rewards associated to some common actions collected by both the demonstration and exploration agents. Still on the HalfCheetah-v2 example, the demonstration agent will reward expert actions at the beginning of the agent run with the high demonstration reward, while the exploration agent will receive poor reward for the same exact actions. This induces a sort of discrepancy between data coming from the offline demonstration agent and experiences coming from the online RL exploration agent. It also implies an overestimation of demonstration actions. However, allocating high reward to demonstration data which are not correlated with the actual reward of the environment might encourage the agent to get to states closer to the expert's ones. Nonetheless, it is difficult to assess the impact on the training in practice.

V. CONCLUSION

We present GRI, a method that efficiently leverages both expert demonstrations and environment exploration. GRI is

straightforward to implement over any off-policy deep reinforcement learning algorithm. GRI-based algorithms improve data efficiency compared to vanilla reinforcement algorithms and do not suffer from distribution shift as much as imitation learning methods. This method also proved to be robust to noisy demonstrations in the expert dataset. We applied GRI to autonomous driving with the distributed GRIAD algorithm and outperformed the previous camera-based state-of-the-art on the CARLA Leaderboard. Finally, we showed its generalizability using different DRL backbones on several Mujoco continuous control environments and highlighted its robustness. In future work, we plan on focusing on LiDAR and camera fusion for GRIAD, as it recently showed to significantly improve the driving quality on CARLA.

REFERENCES

[1] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016. [Online]. Available: http://arxiv.org/abs/1604.07316

[2] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, 2018.

[3] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[4] 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 *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3613–3619.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: http://www.nature.com/articles/nature14236

[6] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms." *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1707.html#SchulmanWDRK17

[7] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 1587–1596. [Online]. Available: https://proceedings.mlr.press/v80/fujimoto18a.html

[8] 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 *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1928–1937. [Online]. Available: https://proceedings.mlr.press/v48/mniha16.html

[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16, license: CC-BY.

[10] D. Chen, V. Koltun, and P. Krähenbühl, "Learning to drive from a world on rails," in *ICCV*, 2021.

[11] F. Codevilla, E. Santana, A. Lopez, and A. Gaidon, "Exploring the Limitations of Behavior Cloning for Autonomous Driving," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 9328–9337. [Online]. Available: https://ieeexplore.ieee.org/document/9009463/

[12] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[13] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Conference on Robot Learning (CoRL)*, 2019.

[14] D. Gordon, A. Kadian, D. Parikh, J. Hoffman, and D. Batra, "SplitNet: Sim2Sim and Task2Task Transfer for Embodied Visual Navigation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 1022–1031. [Online]. Available: https://ieeexplore.ieee.org/document/9009082/

[15] M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[16] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool, "End-to-end urban driving by imitating a reinforcement learning coach," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[17] T. Hester, M. Vecerík, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, A. Sendonaris, G. Dulac-Arnold, I. Osband, J. P. Agapiou, J. Z. Leibo, and A. Gruslys, "Learning from demonstrations for real world reinforcement learning," *CoRR*, vol. abs/1704.03732, 2017. [Online]. Available: http://arxiv.org/abs/1704.03732

[18] S. Reddy, A. D. Dragan, and S. Levine, "SQIL: imitation learning via regularized behavioral cloning," *CoRR*, vol. abs/1905.11108, 2019. [Online]. Available: http://arxiv.org/abs/1905.11108

[19] A. Rajeswaran, V. Kumar, A. Gupta, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *CoRR*, vol. abs/1709.10087, 2017. [Online]. Available: http://arxiv.org/abs/1709.10087

[20] J. B. Martin, R. Chekroun, and F. Moutarde, "Learning from demonstrations with sacr2: Soft actor-critic with reward relabeling," *arXiv preprint arXiv:2110.14464*, 2021.

[21] D. Xu, S. Nair, Y. Zhu, J. Gao, A. Garg, L. Fei-Fei, and S. Savarese, "Neural Task Programming: Learning to Generalize Across Hierarchical Tasks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD: IEEE, May 2018, pp. 3795–3802. [Online]. Available: https://ieeexplore.ieee.org/document/8460689/

[22] Y. Gao, H. Xu, J. Lin, F. Yu, S. Levine, and T. Darrell, "Reinforcement learning from imperfect demonstrations," *CoRR*, vol. abs/1802.05313, 2018. [Online]. Available: http://arxiv.org/abs/1802.05313

[23] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018. [Online]. Available: https://openreview.net/forum?id=HJjvxl-Cb

[24] 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 *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1509.02971

[25] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. G. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," *CoRR*, vol. abs/1710.02298, 2017. [Online]. Available: http://arxiv.org/abs/1710.02298

[26] W. Dabney, G. Ostrovski, D. Silver, and R. Munos, "Implicit quantile networks for distributional reinforcement learning," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 1096–1105. [Online]. Available: https://proceedings.mlr.press/v80/dabney18a.html

[27] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 6105–6114. [Online]. Available: https://proceedings.mlr.press/v97/tan19a.html

[28] M. Toromanoff, E. Wirbel, and F. Moutarde, "Is Deep Reinforcement Learning Really Superhuman on Atari?" in *Deep Reinforcement Learning Workshop of 39th Conference on Neural Information Processing Systems (Neurips'2019)*, Vancouver, Canada, Dec. 2019. [Online]. Available: https://hal-mines-paristech.archives-ouvertes.fr/hal-02368263

[29] Y. Fujita, P. Nagarajan, T. Kataoka, and T. Ishikawa, "Chainerrl: A deep reinforcement learning library," *Journal of Machine Learning Research*, vol. 22, no. 77, pp. 1–14, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-376.html