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

Raphael Chekroun, Marin Toromanoff, Hornauer Sascha, Fabien Moutarde

robotics

MDPI

Article

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

Raphael Chekroun [1,2,*], Marin Toromanoff [2], Sascha Hornauer [1] and Fabien Moutarde [1]

[1] Center for Robotics, Mines Paris, PSL University, 75006 Paris, France
[2] Valeo Driving Assistant Research, 75017 Paris, France
* Correspondence: raphael.chekroun@minesparis.psl.eu

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 agent. 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 method, by 17%.

Keywords: deep reinforcement learning; imitation learning; autonomous driving; Mujoco

Citation: Chekroun, R.; Toromanoff, M.; Hornauer, S.; Moutarde, F. GRI: General Reinforced Imitation and Its Application to Vision-Based Autonomous Driving. *Robotics* **2023**, *12*, 127. https://doi.org/10.3390/robotics12050127

Academic Editors: Luis Payá, Oscar Reinoso García and Helder Jesus Araújo

Received: 18 July 2023
Revised: 28 August 2023
Accepted: 2 September 2023
Published: 6 September 2023

1. Introduction

Autonomous driving (AD) in urban areas is a convoluted task. Agents have to efficiently analyze 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 the literature in autonomous driving focuses on different learning methods rather than the design of general hand-crafted rules.

Imitation learning (IL) [1–4], 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 these 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.

DRL [5–8] 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 GRI, a novel method which combines exploration and demonstration through distillation of prior

knowledge from expert behavior in a classical online RL training. GRI is based on the simplifying hypothesis that expert data present 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 Figure 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 [3,10]. 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].

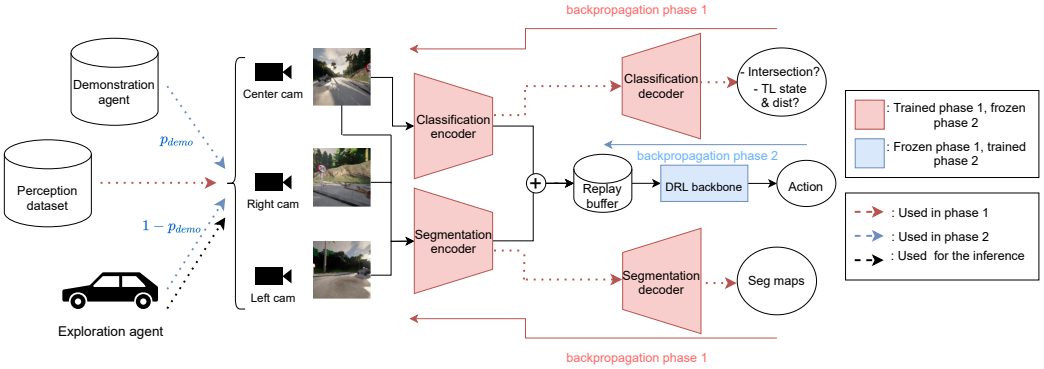

Figure 1. GRI is applied to visual-based autonomous driving in an end-to-end pipeline composed of a perception module encoding RGB images from three cameras on the driving agent and a decision-making module inferring an action from the encoded features. This pipeline is trained in two phases: (1) Visual encoders are pretrained on 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.

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 algorithms 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 the visual-based Autonomous Driving (GRIAD) algorithm.
- Further analysis of GRI-based algorithms on the Mujoco benchmark.

2. 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.

2.1. 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 designed and trained 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.

2.2. 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 [17–20]. In particular, demonstrations can be used to initialize policies by pretraining the network [17,19,21] 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] completes IL using an RL agent. To do so, the replay buffer is initially filled with demonstrations, associated with a constant reward $r_{demo} = 1$. An 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 an 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 the divergent losses issue. Moreover, DQfD, contrarily to GRI, relies on the true environment reward for the expert data, which cannot always be obtained.

3. General Reinforced Imitation

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.

3.1. 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.

We implemented offline *demonstration agents* which sends expert data associated with the reward r_{demo} to the memory buffer. These agents collect transitions from an expert dataset and work concurrently and indistinguishably with exploration agents connected with the simulator to collect states, actions, and rewards.

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.

Algorithm 1: GRI: General Reinforced Imitation.

Input: r_{demo} demonstration reward value, p_{demo} probability to use demonstration agent;
Initialize empty buffer $\mathcal{B}$;
while *not converged* **do**
  if $len(\mathcal{B}) \geq min\_buffer$ **then**
    do a DRL network update;
  end
  if $random.random() \geq p_{demo}$ **then**
    collect episode $(s_t^{online}, a_t, r_t, s_{t+1}^{online})_t$ in buffer $\mathcal{B}$ with exploration agent
  else
    add episode $(s_t^{offline}, a_t, r_{demo}, s_{t+1}^{offline})_t$ in buffer $\mathcal{B}$ with demonstration agent;
  end
end

3.2. 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 to convergence two visual encoders on segmentation and classifications tasks with different camera perspectives to extract compact semantic features.

We found that a single-camera setup leads 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 $x = 2.5$ m, $z = 1.2$ m 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 Figure 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.

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, similar to, in Toromanoff et al. [15], while reducing the encoder latent space dimension by a factor of $\sim$5.

The visual part for the CARLA Leaderboard has been trained on a dataset of 400,000 samples, which corresponds to 44 h 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. Furthermore, 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.

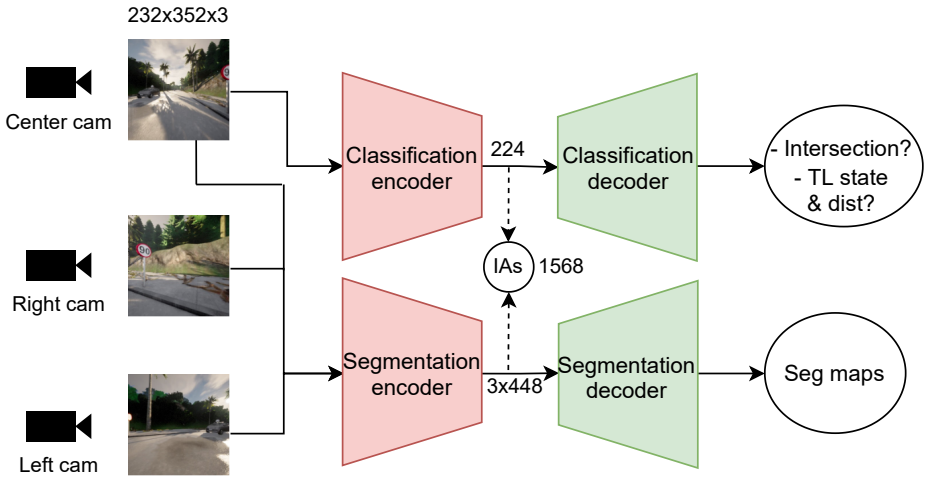

Figure 2. Feature extraction from RGB camera images for the visual subsystem. Two encoder-decoder networks are pretrained on segmentation, classifications, and regression tasks. Classifications and regression are only performed on the center image while all three images are segmented. After training, 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.

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 Figure 3.

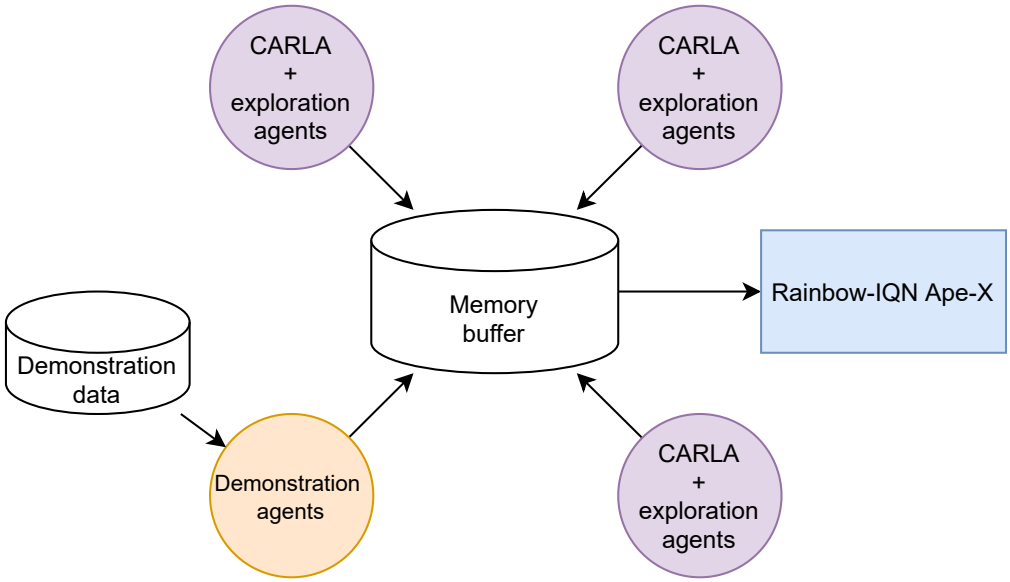

Figure 3. Simplified representation of the distributed GRIAD setup with a Rainbow-IQN Ape-X backbone. A central computer receives data in a shared replay buffer from both exploration and demonstration agents running on other computers. Data are sampled from this replay buffer to make the backpropagation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

Demonstration agents send to the memory buffer samples of expert trajectory from the demonstration dataset. This dataset consists of 22 h of driving, which correspond to 200,000 samples, generated using the autopilot from CARLA on predefined tracks published by CARLA. 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, ∼10% of our demonstrations correspond to poor action choices. However, we decided to use this demonstration dataset as 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 $r_{demo} = 1$.

4. 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.

4.1. GRIAD on CARLA

On the CARLA leaderboard. We trained GRIAD for 60 M steps (∼45 M exploration steps +200,000 expert data sampled ∼15 M times). Both visual and decision-making parts were trained on all available maps with all available weather. We compare the top three of camera-based and LiDAR-based methods, also distinguishing methods exploiting or not the Inertial Movement Unit (IMU) sensor, on the CARLA Leaderboard.

Our method outperforms World on Rails, the previous comparable leading method on the CARLA leaderboard, by ∼17% on the main metric, the driving score, while using fewer sensors.

However, more recent LiDAR-based methods, or methods exploiting IMU sensor, give significantly better results but cannot be compared directly as inputs are of a different nature. Indeed, LiDAR allows accurate depth measurement, which allows an increase of accuracy on vision related task after fusion with camera data [29]. IMU sensor replaces approximate orientation estimation by precise measurement, therefore providing accurate positional information. Therefore, using LiDAR and/or IMU sensors on top of camera leads to richer input features for deep learning models. In this work, we chose to build an autonomous driving system using camera only, as it leads to less expensive and complex systems. GRIAD pipeline can be augmented with other sensors by changing the vision subsystem.

CARLA Leaderboard results are presented in Table 1.

Ablation study on the NoCrash benchmark. We provide an ablation study on the demonstration agents in the GRIAD setup. We compare GRIAD trained with nine exploration agents and three demonstration agents, to GRIAD trained with nine explorations agents, i.e., regular RL on the NoCrash benchmark [11].

Agents are trained on a single environment (Town01) under a specific set of training weather. They are then evaluated on several scenarios with different traffic density on the training (Town01) and test (Town02) town with training and test sets of weather.

Table 1. Top three of camera-based and LiDAR-based agents with and without IMU sensors on the CARLA Leaderboard on August 2023. Results of reproduced methods are not considered. Driving metrics are: driving score (DS, main metric), route completion (RC), and infraction score (IS). Higher is better for all metrics. Our method improves the driving score by 17% relative to the prior camera-based IMU-less state-of-the-art method [10], while using fewer cameras than the two other best methods in this category.

Method	Cam.	LiDAR	IMU	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
TCP [30]	1	✗	✓	**75.13**	**85.53**	**0.87**
Latent Transfuser [3]	3	✗	✓	45.2	66.31	0.72
LBC [13]	3	✗	✓	10.9	21.3	0.55
ReasonNet [31]	4	✓	✓	**79.95**	89.89	**0.89**
LAV [32]	4	✓	✓	61.8	**94.5**	0.64
InterFuser [33]	3	✓	✓	76.18	88.23	0.84
Transfuser+ [3]	4	✓	✗	50.5	73.8	0.68

For these experiments, GRIAD was trained on 16M samples corresponding to 12 M 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 12 M exploration steps and the other on 16 M 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 2.

Table 2. Ablation study of GRIAD using the NoCrash benchmark. Mean and standard deviation are computed over three evaluation seeds. Score is the percentage of road completed without any crash. Explo. xM + Demo. yM means the network has been trained on x million samples from exploration agents and y million samples from demonstration agents. GRIAD leveraging only exploration agents is regular RL. GRIAD experimentally generalizes more on test weather than RL trained on 12 M and 16 M exploration samples and globally gives the best agent. GRIAD trained with demonstration agents only leads to scores of 0 on every task, as every sample has the same reward during the training.

Task	Town, Weather	GRIAD		
		Explo. 12 M	Explo. 12 M + Demo. 4 M	Explo. 16 M
Empty	train, train	96.3 ± 1.5	**98.0 ± 1.7**	**98.0 ± 1.0**
Regular		95.0 ± 2.4	**98.3 ± 1.7**	**98.6 ± 1.2**
Dense		91.7 ± 2.0	93.7 ± 1.7	**95.0 ± 1.6**
Empty	test, train	83.3 ± 3.7	94.0 ± 1.6	**96.3 ± 1.7**
Regular		82.6 ± 3.7	93.0 ± 0.8	**96.3 ± 2.5**
Dense		61.6 ± 2.0	**77.7 ± 4.5**	**78.0 ± 2.8**
Empty	train, test	67.3 ± 1.9	**83.3 ± 2.5**	73.3 ± 2.5
Regular		76.7 ± 2.5	**86.7 ± 2.5**	81.3 ± 2.5
Dense		67.3 ± 2.5	**82.6 ± 0.9**	80.0 ± 1.6
Empty	test, test	60.6 ± 2.5	**68.7 ± 0.9**	62.0 ± 1.6
Regular		59.3 ± 2.5	**63.3 ± 2.5**	56.7 ± 3.4
Dense		40.0 ± 1.6	**52.0 ± 4.3**	46.0 ± 3.3

We first observe that GRIAD systematically gives better results than RL with 12 M steps, while taking approximately the same time to train (+∼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 ∼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. The 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 a score of 0 on every evaluated task. 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.

4.2. GRI on the Mujoco Benchmark

To further validate the GRI method, we conducted experiments on selected Mujoco [12] environments, shown in Figure 4. Expert data were generated using chainerrl [34] pre-trained 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 are not always significantly better than our trained vanilla RL network. Hence, this study assesses the efficiency of GRI even with suboptimal expert data.

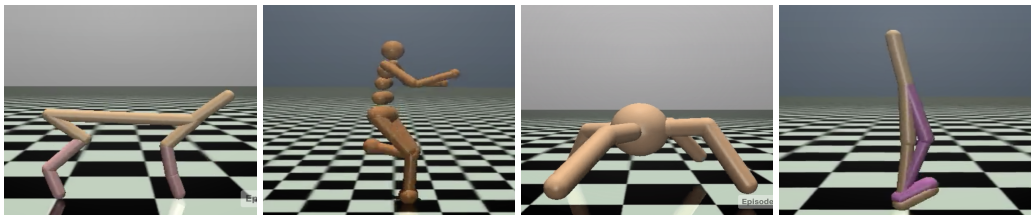

Figure 4. Mujoco environments used for our experiments. Respectively HalfCheetah-v2, Humanoid-v2, Ant-v2, and Walker2d-v2. Articulations are controlled to make them walk. Rewards depend on the covered distance.

Study on the proportion of demonstration agents. Since the experiments were 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 agents between 0% and 40%. Each experiment has been repeated three times, with different seeds. Figure 5 presents the results with the variances and the evaluation reward of the expert. Experiments were conducted with public code from GitHub (original code from https://github.com/dongminlee94/deep_rl, accessed on 3 November 2021), 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 4.3. However, the rewards turns out to become significantly higher after some time. Here, GRI-SAC is better than SAC with every proportion of demonstration agents. The 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. The 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. Therefore, we conclude that the proportion of demonstration agent should not exceed 50%. We discuss some qualitative insights in Section 4.3.

These experiments reveal, 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.

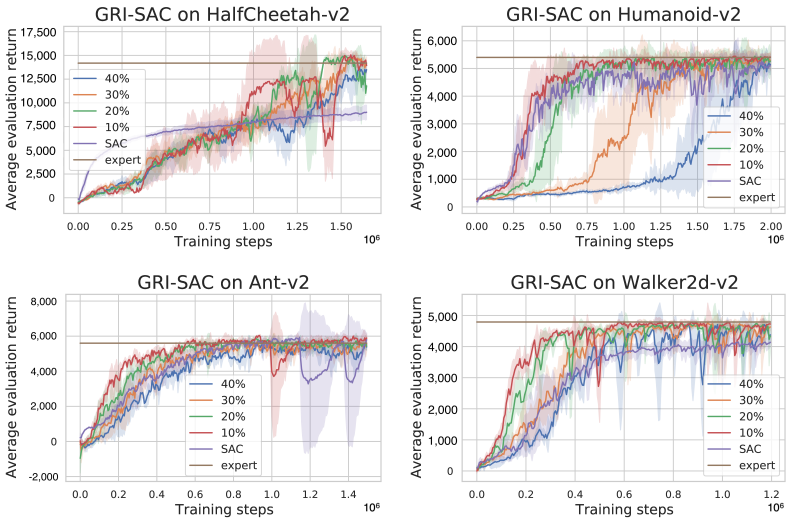

Figure 5. Ablation over demonstration agents with the GRI-SAC setup on Mujoco environments, and analysis of the evolution of the evaluation reward in function of the proportion of demonstration agents. GRI-SAC with 0% demonstration agent is vanilla SAC. We observe that GRI-SAC always reaches the level of the expert even when the expert is significantly better than the trained vanilla SAC. The proportion of demonstration agent has a significant impact on the dynamics of the convergence.

GRI with DDPG as the 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 Figure 6.

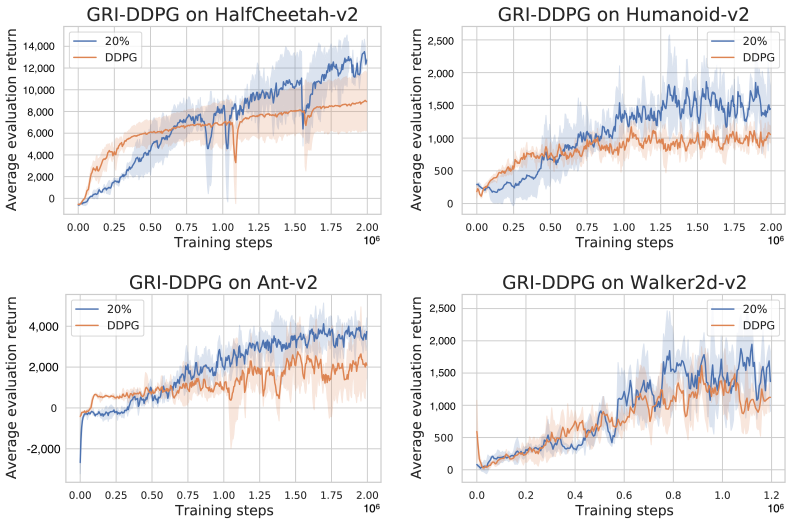

Figure 6. Ablation over demonstration agents with the GRI-DDPG, with 20% of demonstration agents on Mujoco environments. GRI-DDPG systematically leads to a better reward than vanilla DDPG. However, contrary to GRI-SAC, GRI-DDPG with 20% demonstration agents does not systematically reach the expert level.

We observe that, similar 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 (compare Figure 5). We can conclude that GRI is easily adaptable and generalizes to locomotion tasks, where it robustly outperforms the two alternative methods.

4.3. Limitations and Quantitative Insights

The main limitations of this method are the 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 are 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 Figure 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. However, to reach a state where it can successfully jump, it needs to warm up to gain the required speed and momentum by doing some low reward actions. Hence, our GRI-SAC agent learns to jump before it is able to walk, making it fall. Once the agent learned how to reach the jumping state, rewards 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 for the distribution shift between demonstration and exploration data.

Finally, a third limitation of our approach is the inconsistency of the rewards associated with some common actions collected by both the demonstration and exploration agents. Still for 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 ones. Nonetheless, it is difficult to assess the impact on the training in practice.

5. Conclusions

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 method 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.

Author Contributions: Conceptualization, R.C.; methodology, R.C.; software, R.C.; validation, R.C.; formal analysis, R.C.; investigation, R.C.; resources, F.M.; data curation, R.C.; writing—original draft preparation, R.C.; writing—review and editing, R.C., M.T., S.H. and F.M.; visualization, R.C.; supervision, M.T., S.H. and F.M.; project administration, R.C.; funding acquisition, F.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The simulators used for this research are public and available online. CARLA simulator: https://github.com/carla-simulator/carla, accessed on 7 February 2021 and Mujoco simulator: https://github.com/deepmind/mujoco, accessed on 2 November 2021.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bojarski, M.; Testa, D.D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L.D.; Monfort, M.; Muller, U.; Zhang, J.; et al. End to End Learning for Self-Driving Cars. *arXiv* **2016**, arXiv:1604.07316.
2. Osa, T.; Pajarinen, J.; Neumann, G.; Bagnell, J.A.; Abbeel, P.; Peters, J. An algorithmic perspective on imitation learning. *Found. Trends® Robot.* **2018**, *7*, 1–179 . [CrossRef]
3. Prakash, A.; Chitta, K.; Geiger, A. Multi-Modal Fusion Transformer for End-to-End Autonomous Driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021.
4. Toromanoff, M.; Wirbel, E.; Wilhelm, F.; Vejarano, C.; Perrotton, X.; Moutarde, F. End to End Vehicle Lateral Control Using a Single Fisheye Camera. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 3613–3619. [CrossRef]
5. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
6. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347.
7. Fujimoto, S.; van Hoof, H.; Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 1587–1596.
8. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous Methods for Deep Reinforcement Learning. In Proceedings of the The 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; Volume 48, pp. 1928–1937.
9. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An Open Urban Driving Simulator. In Proceedings of the 1st Annual Conference on Robot Learning, Mountain View, CA, USA, 13–15 November, 2017; pp. 1–16.
10. Chen, D.; Koltun, V.; Krähenbühl, P. Learning to drive from a world on rails. In Proceedings of the ICCV, Virtual, 11–17 October 2021.
11. Codevilla, F.; Santana, E.; Lopez, A.; Gaidon, A. Exploring the Limitations of Behavior Cloning for Autonomous Driving. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9328–9337. [CrossRef]
12. Todorov, E.; Erez, T.; Tassa, Y. MuJoCo: A physics engine for model-based control. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 5026–5033. [CrossRef]
13. Chen, D.; Zhou, B.; Koltun, V.; Krähenbühl, P. Learning by Cheating. In Proceedings of the Conference on Robot Learning (CoRL), London, UK, 8–11 November 2019.
14. Gordon, D.; Kadian, A.; Parikh, D.; Hoffman, J.; Batra, D. SplitNet: Sim2Sim and Task2Task Transfer for Embodied Visual Navigation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1022–1031. [CrossRef]
15. Toromanoff, M.; Wirbel, E.; Moutarde, F. 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), Seattle, WA, USA, 13–19 June 2020.
16. Zhang, Z.; Liniger, A.; Dai, D.; Yu, F.; Van Gool, L. End-to-End Urban Driving by Imitating a Reinforcement Learning Coach. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual, 11–17 October 2021.
17. Hester, T.; Vecerík, M.; Pietquin, O.; Lanctot, M.; Schaul, T.; Piot, B.; Sendonaris, A.; Dulac-Arnold, G.; Osband, I.; Agapiou, J.P.; et al. Learning from Demonstrations for Real World Reinforcement Learning. *arXiv* **2017**, arXiv:1704.03732.
18. Reddy, S.; Dragan, A.D.; Levine, S. SQIL: Imitation Learning via Regularized Behavioral Cloning. *arXiv* **2019**, arXiv:1905.11108.
19. Rajeswaran, A.; Kumar, V.; Gupta, A.; Schulman, J.; Todorov, E.; Levine, S. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. *arXiv* **2017**, arXiv:1709.10087.
20. Martin, J.B.; Chekroun, R.; Moutarde, F. Learning from demonstrations with SACR2: Soft Actor-Critic with Reward Relabeling. *arXiv* **2021**, arXiv:2110.14464.

21. Xu, D.; Nair, S.; Zhu, Y.; Gao, J.; Garg, A.; Fei-Fei, L.; Savarese, S. Neural Task Programming: Learning to Generalize Across Hierarchical Tasks. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 3795–3802. [CrossRef]
22. Gao, Y.; Xu, H.; Lin, J.; Yu, F.; Levine, S.; Darrell, T. Reinforcement Learning from Imperfect Demonstrations. *arXiv* **2018**, arXiv:1802.05313.
23. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
24. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. In Proceedings of the 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May 2016.
25. Hessel, M.; Modayil, J.; van Hasselt, H.; Schaul, T.; Ostrovski, G.; Dabney, W.; Horgan, D.; Piot, B.; Azar, M.G.; Silver, D. Rainbow: Combining Improvements in Deep Reinforcement Learning. *arXiv* **2017**, arXiv:1710.02298.
26. Dabney, W.; Ostrovski, G.; Silver, D.; Munos, R. Implicit Quantile Networks for Distributional Reinforcement Learning. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Volume 80, pp. 1096–1105.
27. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 6105–6114.
28. Toromanoff, M.; Wirbel, E.; Moutarde, F. Is Deep Reinforcement Learning Really Superhuman on Atari? In Proceedings of the Deep Reinforcement Learning Workshop of 39th Conference on Neural Information Processing Systems (Neurips'2019), Vancouver, BC, Canada, 8–14 December 2019.
29. Hu, H.; Liu, Z.; Chitlangia, S.; Agnihotri, A.; Zhao, D. Investigating the impact of multi-lidar placement on object detection for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 2550–2559.
30. Wu, P.; Jia, X.; Chen, L.; Yan, J.; Li, H.; Qiao, Y. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 6119–6132.
31. Shao, H.; Wang, L.; Chen, R.; Waslander, S.L.; Li, H.; Liu, Y. ReasonNet: End-to-End Driving with Temporal and Global Reasoning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 13723–13733.
32. Chen, D.; Krähenbühl, P. Learning from all vehicles. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 17222–17231.
33. Shao, H.; Wang, L.; Chen, R.; Li, H.; Liu, Y. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In Proceedings of the Conference on Robot Learning, Atlanta, GA, USA, 6–9 November 2023; pp. 726–737.
34. Fujita, Y.; Nagarajan, P.; Kataoka, T.; Ishikawa, T. ChainerRL: A Deep Reinforcement Learning Library. *J. Mach. Learn. Res.* **2021**, *22*, 3557–3570.