

---

# Stable and Efficient Reinforcement Learning Method for Avoidance Driving of Unmanned Vehicles

---

Sun-Ho Jang , [Dong-Sung Pae](#) , [Woo-Jin Ahn](#) , Yu-Jin Kim , Hyung-Gil Hong , [Myo-Taeg Lim](#) \*

Posted Date: 10 August 2023

doi: 10.20944/preprints202308.0756.v1

Keywords: reinforcement learning ; meta learning ; deep reinforcement learning ; autonomous driving ; robot operating system



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Article

# Stable and Efficient Reinforcement Learning Method for Avoidance Driving of Unmanned Vehicles

Sun-Ho Jang<sup>1,2</sup> , Woo-Jin Ahn<sup>2</sup> , Yu-Jin Kim<sup>2</sup>, Hyung-Gil Hong<sup>1</sup>, Dong-Sung Pae<sup>3</sup> , Myo-Taeg Lim<sup>2</sup> \*

<sup>1</sup> Korea Institute of Robotics Technology Convergence, Andong 36728, Korea; jang1229@kiro.re.kr (S.-H.J.); honghg@kiro.re.kr (H.-G.H.)

<sup>2</sup> School of Electrical Engineering, Korea University, Seoul 02841, Korea; wjahn@korea.ac.kr (W.-J.A.); sally1004k@korea.ac.kr (Y.-J.K.)

<sup>3</sup> Department of Software, Sangmyung University, Cheonan 31066, Korea

\* Correspondence: mlim@korea.ac.kr (M.T.L.); paeds915@smu.ac.kr (D.S.P)

**Abstract:** Reinforcement Learning (RL) has demonstrated considerable potential in solving challenges across various domains, notably in autonomous driving. Nevertheless, implementing RL in autonomous driving comes with its own set of difficulties, such as the overestimation phenomenon, extensive learning time, and sparse reward problems. Although solutions like Hindsight Experience Replay (HER) have been proposed to alleviate these issues, the direct utilization of RL in autonomous vehicles remains constrained due to the intricate fusion of information and the possibility of system failures during the learning process. In this paper, we present a novel RL-based autonomous driving system technology that combines Obstacle Dependent Gaussian (ODG) RL, Soft Actor-Critic (SAC), and meta-learning algorithms. Our approach addresses key issues in RL, including the overestimation phenomenon and sparse reward problems, by incorporating prior knowledge derived from the ODG algorithm. We evaluated our proposed algorithm on official F1 circuits, using high-fidelity racing simulations with complex dynamics. The results demonstrate exceptional performance, with our method achieving up to 89% faster learning speed compared to existing algorithms in these environments.

**Keywords:** reinforcement learning; meta learning; deep reinforcement learning; autonomous driving; robot operating system

## 1. Introduction

Reinforcement Learning (RL) has recently gained notable attention in various fields, including autonomous driving, due to its capability to address unanticipated challenges in real-world scenarios. Autonomous driving systems, by employing RL algorithms, are able to accrue experience and refine their decision-making procedures within dynamic environments.[1–3] This can be largely attributed to RL's inherent ability to adapt and learn from complex and fluctuating situations, demonstrating its aptitude for these applications. The basis concept of RL lies in the structure of Markov Decision Processes (MDP), a system where algorithms learn via a trial-and-error approach, striving to reach predetermined objectives by learning from mistakes and rewards. The aim of RL is to optimize future cumulative rewards and formulate the most efficient policy for distinct problems [4]. Recent integration of deep learning with RL has demonstrated promising outcomes across various domains. This involves the employment of advanced neural networks such as Convolutional Neural Networks (CNNs), multi-layer perceptrons, restricted Boltzmann machines, and recurrent neural networks[5,6]. By fusing reinforcement learning with deep learning, the system's learning capabilities are significantly enhanced, allowing it to process complex data such as sensor feedback and environmental observations, thus facilitating more informed and effective driving decisions [7]. However, the application of RL to autonomous driving presents a unique array of challenges, particularly when it comes to deploying RL in real-world environments. The uncertainties inherent in these environments can make the effective

execution of RL quite challenging. As a result, researchers often struggle to achieve optimal RL performance directly within the actual driving context, highlighting the various obstacles encountered when applying RL to autonomous driving[8]. Several challenges plague the application of RL to autonomous driving: overestimation phenomenon, learning time, and sparse reward problems[9,10].

Firstly, the overestimation phenomenon is prevalent in model-free RL methods, such as Q-learning[11] and its variants like Double Deep Q Network (DDQN)[12,13] and Dueling DQN[14]. These methods are susceptible to overestimation and incorrect learning, primarily due to the combination of insufficiently flexible function approximation and the presence of noise, which lead to inaccuracies in action values. Secondly, the significant amount of learning time required is another hurdle. When RL is fused with neural networks, it generates policies directly from interactions with the environment, bypassing the need for a basic dynamics model. However, even simple tasks necessitate extensive trials and a massive amount of data for learning. This makes high-performance RL both time-consuming and data-intensive [15]. Lastly, the issue of sparse reward arises during RL training. This presents challenges in scenarios where not all conditions receive immediate compensation. Although techniques like Hindsight Experience Replay (HER)[16,17] have been proposed to mitigate this issue, the direct application of RL to autonomous vehicles is still limited due to the complex fusion of information and potential system failures during the learning process. This paper addresses the challenges of RL in autonomous driving and reduces the reliance on extensive real-world learning by introducing a set of innovative techniques to enhance the efficiency and effectiveness of RL : data preprocessing through Obstacle Dependent Gaussian (ODG) [18,19] DQN, prior knowledge through Guide ODG DQN, and meta-learning-based guided ODG DDQN.

The data preprocessing method employs the ODG algorithm to combat the overestimation phenomenon. By preprocessing distance information through ODG DQN, it allows for more accurate action values, fostering stable and efficient learning [20]. The prior knowledge method draws on human learning mechanisms, incorporating knowledge derived from the ODG algorithm. This strategy mitigates the issue of sparse rewards and boosts the learning speed [21], facilitating more effective convergence. Lastly, the meta-learning-based guide rollout method uses ODG DQN to address complex driving decisions and sparse rewards in real-world situations. By enriching prior knowledge using a rollout approach, this method aims to create efficient and successful autonomous driving policies.

Our main contributions can be summarized as follows :

- **Efficiency and Speed of Learning:** The newly proposed RL algorithm utilizes ODG DQN on preprocessed information, enabling the agent to make optimal action choices, which significantly enhances the learning speed and efficiency.
- **Improvement of Learning Stability:** With the use of prior knowledge, the Guide-ODG-DQN helps mitigate the issue of sparse rewards, thus increasing the learning stability and overall efficiency.
- **Adaptability to Various Environments:** The Meta-learning-based ODG DDQN leverages model similarities and differences to increase learning efficiency. This allows for the reliable training of a universal model across diverse environments, with its performance demonstrated in environments like Gazebo and Real-Environment.

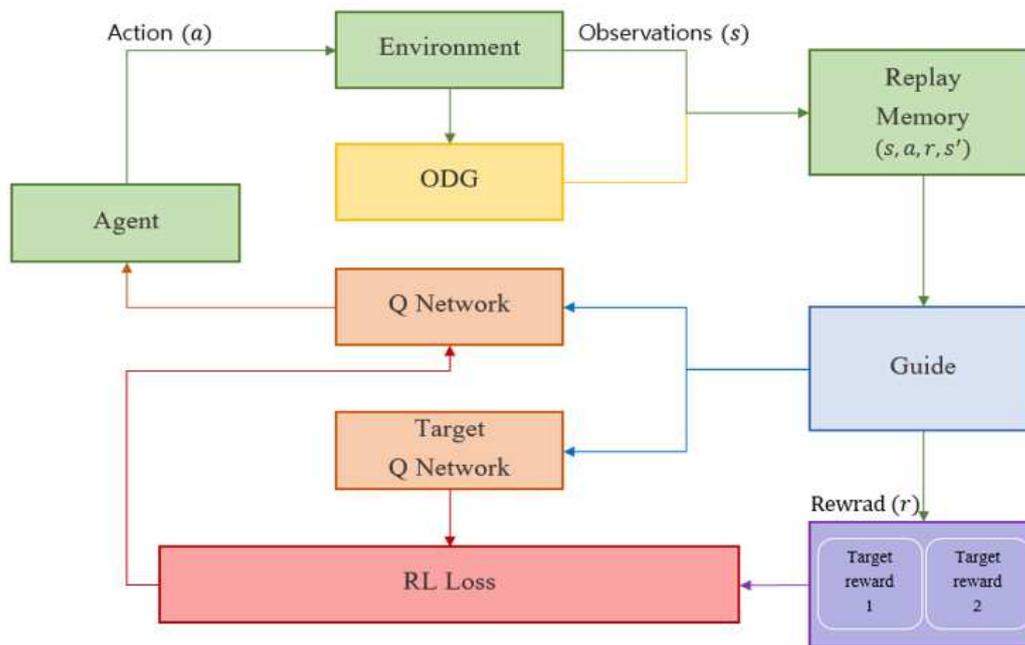
The remainder of this paper is organized as follows: In the stable and efficient method section, we mainly introduce the proposed reinforcement learning algorithm. To verify the effectiveness of our work, the experimental evaluations and necessary analysis are presented in experiment. Finally, we summarize our work in Conclusions.

## 2. Stable and Efficient Reinforcement Learning Method

LiDAR information serves as an invaluable perspective for autonomous driving systems, functioning much like a driver's sense by identifying obstacles through environmental analysis. LiDAR-based RL methods have found extensive application in research focused on judgement and

control within autonomous driving systems such as Partially observable Markov Decision Process (POMDP) [22]. However, learning methodologies based on Q-learning, such as DDQN, encounter persistent overestimation issues, posing obstacles to the enhancement of learning efficiency and convergence speed.

To mitigate these issues, we propose a method which preprocesses and transforms the LiDAR value into valuable information attuned to the operating environment, implementing it as the ODG technique [23]. This approach, as depicted by the ODG module (in yellow) in Fig. 1, is designed to reduce learning convergence time and boost efficiency by preprocessing RL input data, thus remedying scenarios with inaccurate action values. Furthermore, we introduce the concept of prior knowledge to address the sparse rewards issue that impedes RL's learning stability [24]. By integrating prior knowledge information from sparse reward sections, as demonstrated in the Guide-ODG-DQN framework shown in the guide module (in blue) in Fig. 1, we can enhance learning stability.



**Figure 1.** Process flow of stable and efficient reinforcement learning (SERL).

It's noted that in RL, model performance can decline when the learning environment changes. Thus, we propose the meta-Guide ODG-DDQN method, represented in the target reward module (in purple) in Fig. 1, to devise a more robust and adaptable RL algorithm. After training the model according to an initial goal, we modify the reward function to attain subsequent objectives. This approach effectively communicates the action value to the agent in diverse obstacle environments with reliability and swiftness. The proposed methodology consists of three progressively developed algorithms.

### 2.1. ODG DQN

Overestimation, a consequence of inaccurate action values, is underscored as a critical issue in the DDQN literature. Traditional LiDAR information incorporates an 'Inf' range, which represents all information at the maximum distance or the value of obstacle-free spaces. This arrangement leads to an overlap of LiDAR information within the system, causing overestimation and impeding the model's ability to select these 'Inf' values. In Q-learning, this predicament can be defined by  $Q(s, a) = V_*(s)$  for a given state  $s$ , as detailed in Eq. (1). When environmental noise triggers an error, it is defined per Eq. (2). If the max function is applied at the moment of peak value in Q-learning for action selection, the

expression aligns with Eq. (3). The bias, symbolized by  $\sqrt{C/m-1}$ , causes the model to overestimate the bias relative to the optimal value.

$$\sum_a (Q_t(s, a) - V_*(s)) = 0, \quad (1)$$

$$\frac{1}{m} \sum_a (Q_t(s, a) - V_*(s))^2 = C, \quad (2)$$

$$\max_a Q_t(s, a) \geq V_*(s) + \sqrt{\frac{C}{m-1}}, \quad (3)$$

where  $m$  is the number of actions, and  $C$  is a constant.

To address this overestimation, our algorithm utilizes the ODG module to preprocess state values. Illustrated in Fig. 2, this module, based on Eq. (4), is engineered to establish an optimized steering angle model for the agent via Q-learning-based RL. This paves the way for the development of an optimized path plan built on the steering angle generated by the agent.

$$Q(s_t, a_t) = \mathbb{E}[r_t + \gamma \max_a Q(s_{t+1}, a)]. \quad (4)$$

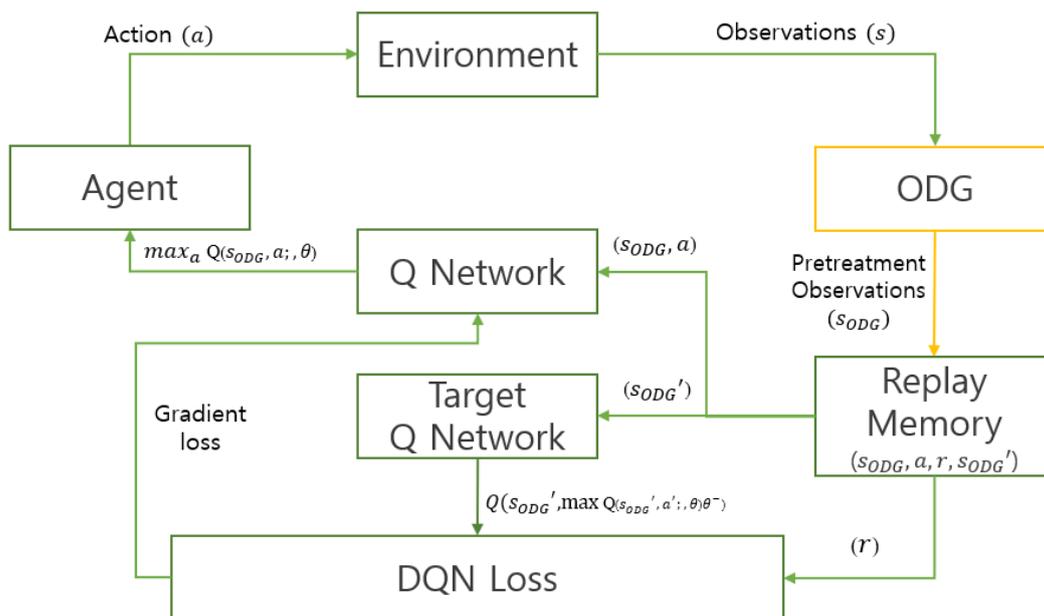
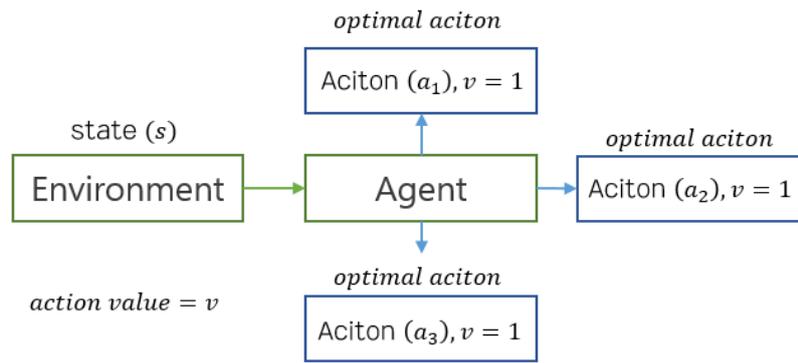


Figure 2. ODG DQN Structure.

LiDAR information, a principal component in autonomous driving systems, is preprocessed via the ODG module, subsequently offering the processed data to the RL approach as the state value. Through the use of a Gaussian distribution, the ODG module converts LiDAR information into continuous values. As depicted in Fig. 3, the creation of a unique state happens when an agent selects an action, preventing the duplication of action values and facilitating a more efficient selection of the optimal action value in accordance with the equation.



**Figure 3.** Overestimation in Q-learning.

For the implementation of our proposed algorithm to RL using LiDAR information, a standard procedure in autonomous vehicles, we employ ODG-based preprocessed LiDAR information. As demonstrated in Fig. 4, the yellow line corresponds to the original LiDAR data, whereas the blue line symbolizes post-processed data. This data includes information on obstacle location and size, derived using Eq. (5) with ODG.

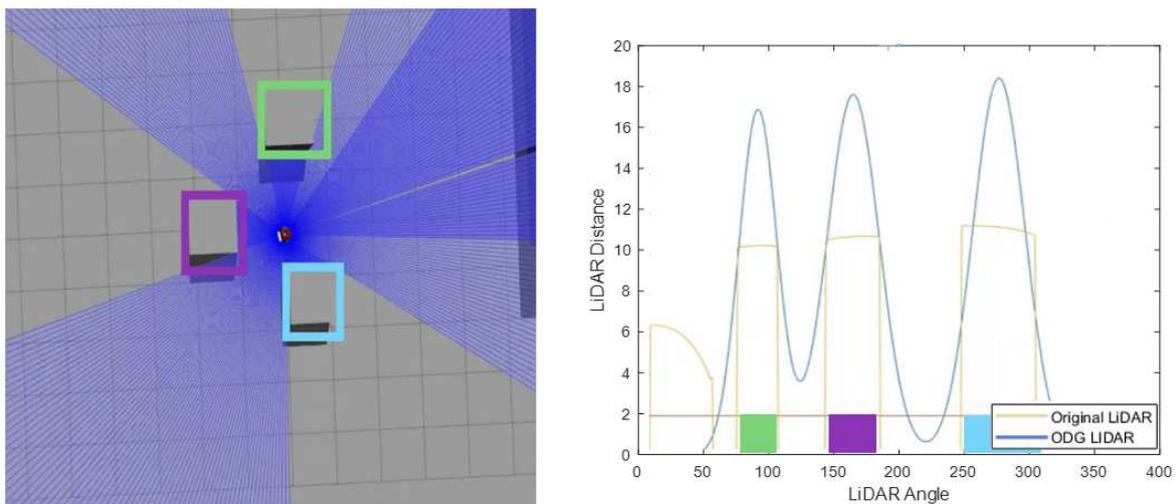
$$a = f_{rep}(\theta_i) = \sum_{k=1}^n A_k \exp\left(-\frac{(\theta_k - \theta_i)^2}{2\sigma_k^2}\right), \quad (5)$$

where

$$A_k = (d_{\max} - d_k) \exp(0.5), \quad (6)$$

$$Q(s_t^\Delta, a_t) = \mathbb{E}[r_t + \gamma \max_a Q(s_{t+1}^\Delta, a)]. \quad (7)$$

In contrast to the overlapping LiDAR information provided by conventional methods, ODG supplies non-overlapping LiDAR data, adjusting the maximum range according to the obstacle's size and distance. This preprocessing enables the agent to make more efficient decisions related to optimal action values based on the processed information, thereby enhancing both the speed and efficiency of learning.



**Figure 4.** Difference between traditional LiDAR and ODG information.

## 2.2. Guide ODG DQN

The Soft Actor Critic (SAC) method [25] is a robust approach that allows observation of multiple optimal values while avoiding the selection of impractical paths. This facilitates a more extensive policy exploration. The SAC employs an efficient and stable entropy framework for the continuous state and action space. As delineated in Eq. (9), the SAC learns the optimal Q function through updating Q-learning via the maximum entropy RL method.

$$\sum_t \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [r(s_t, a_t)], \quad (8)$$

$$J(\pi) = \sum_{t=0}^T \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot | s_t))]. \quad (9)$$

The algorithm initially makes the guide value sparse and, as learning progresses, gradually densifies it, employing the gamma value as outlined in Eq. (11). The term  $\min A$  is representative of the environmental vehicle.

$$\min_{\psi} A < \Delta_{\psi} < \max_{\psi} A, \quad (10)$$

$$v = \max_{\psi} A - |\Delta_{\psi}|. \quad (11)$$

A report on hierarchical deep RL, an approach that implements RL via multiple objectives, emphasized the need to solve sparse reward problems as environments become increasingly diverse and complex. Normally, in problems tackled by RL, rewards are generated for each state, like survival time or score. Every state is linked to an action, receives a reward, and identifies the Q-value so as to maximize the sum of the rewards. However, there are instances where a reward may not be received for each state. These scenarios are referred to as sparse rewards.

$$Q(s_t, a_t) = \mathbb{E}[r_t + \alpha \gamma \mathcal{G} \max_a Q(s_{t+1}, a) + (1 - \alpha) \gamma \max_a Q(s_{t+1}, a)], \quad (12)$$

where

$$\mathcal{G} \max_a f(a) := \text{Guide}_{\text{action}}(S_{t+1}^A). \quad (13)$$

Our proposed solution to these issues is the Guide-ODG-DNQ model that integrates SAC with ODG-DQN. This proposed Guide-ODG-DQN algorithm transforms the initial Q-value from the state value. This value is extracted from the environment, and it is connected with the ODG formula, which is our prior knowledge, and the LiDAR value extracted with ODG, as depicted in Fig. 5. The algorithm extracts a guide action that minimizes the cases where a reward isn't received for every state.

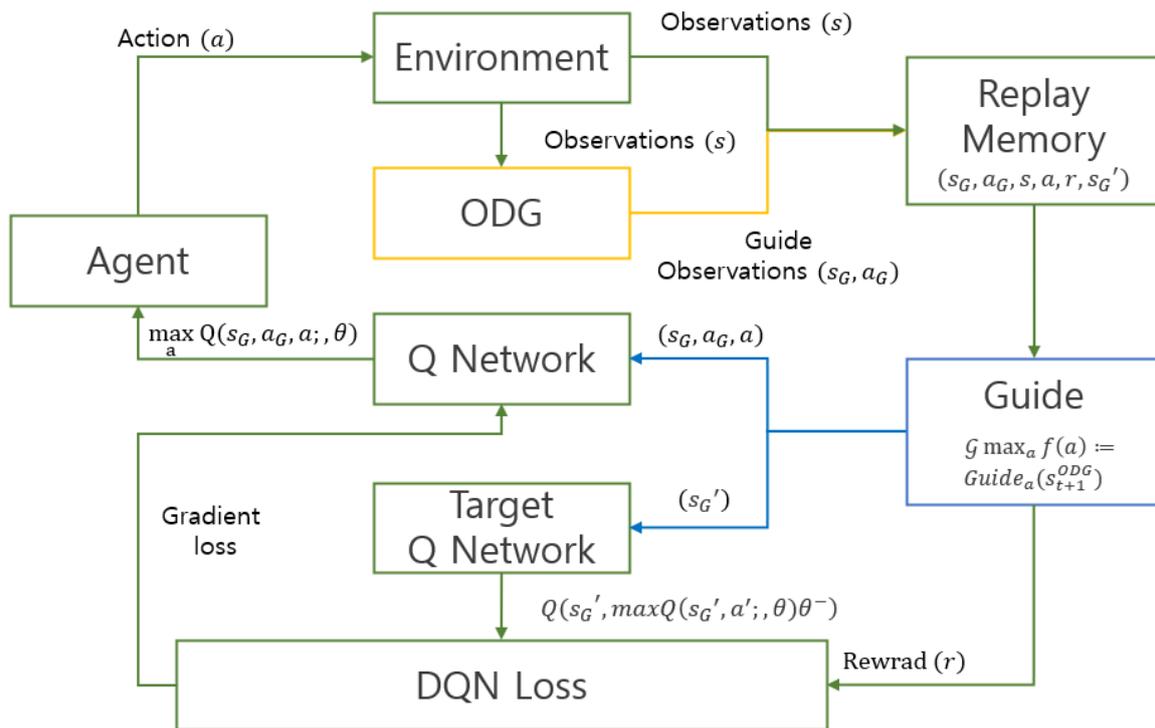


Figure 5. Structure of guide-ODG-DQN.

The Guide-ODG-DQN is designed to store high-quality information values in the replay memory from the outset based on prior knowledge. The agent then continues learning based on this prior knowledge, facilitating easier adaptation to various environments and enabling faster and more stable convergence. Moreover, to prevent an over-reliance on prior knowledge that could compromise the effectiveness of RL, the agent learns from its own experiences during the learning process, which are represented by the gamma value. The agent also contrasts this newly learned information with the values derived from the existing prior knowledge. Consequently, our proposed Guide-ODG-DQN mitigates the sparse reward phenomenon, thereby enhancing both the stability and efficiency of the learning process.

### 2.3. Meta-learning-based Guide ODG DDQN

RL is fundamentally a process of learning through trial and error. The RL agent must experience a diverse set of situations, making decisions in each scenario to understand which actions yield the highest rewards. Striking a balance between experimentation, to ensure no high-reward actions are overlooked, and leveraging acquired knowledge to maximize rewards is crucial. However, achieving this balance typically necessitates numerous trials and, consequently, large volumes of data. Training an RL agent with excessive data might result in overfitting, wherein the agent conforms too closely to the training data and fails to generalize well to new circumstances.

To overcome these limitations, we introduce a novel method known as meta-learning-based Guide-ODG-DDQN. This approach involves storing rewards for each step an integral part of RL in the replay memory, with the stored rewards divided according to the number of targets to be learned as shown in Fig. 6. This model facilitates few-shot learning within RL by training the model to recognize similarities and differences, thus preparing it to perform proficiently in unfamiliar environments with minimal data. The training is guided by two main objectives. The first is to train the target model using the initial reward, while the second is to continue learning by reducing the weight assigned to the initial reward and increasing the weight of the reward for the subsequent target, as depicted in Eq.



chosen with care for distinct aspects of the study: the Gazebo map was used to evaluate learning efficiency and stability, the Sochi map for learning strength, and the Silverstone map to test adaptability to complex conditions, as shown in Fig. 7. The experiment setup was designed to reflect real world dimensions, such that each unit length in the simulation corresponded to one meter in reality [26,27].

First, the index for learning efficiency is determined as follows. As learning begins to converge, the learning efficiency corresponding to the highest reward for the number of frames (in millions) experienced in the virtual environment is considered. Based on the DQN algorithm, we evaluate how fast convergence occurs and how high the reward is.



Figure 7. Map environment.

second the evaluation metric for learning stability assesses how well the path plan generated through RL matches the target path pursued. The path plan created by RL in the virtual environment,  $P_{RL}$ , and path plan created with the ODG,  $P_{ODG}$ , are represented in terms of the  $RMSE$ . Errors between  $P_{ODG}$  and  $P_{RL}$  are computed using Eq. (15).

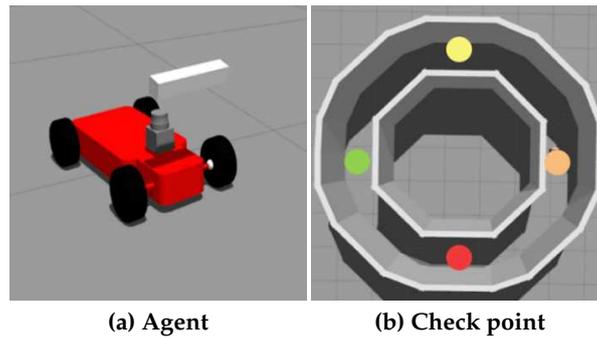
$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}}, \quad (15)$$

where  $\hat{y}_i$  represents the path generated by ODG in  $P_{ODG}$ , which is known to exhibit high real-time performance and stability, and  $y_i$  corresponds to  $P_{RL}$ , which is the path plan generated through RL. The optimal route with the highest reward is considered.  $n$  is the number of steps the agent operates in the simulation environment, corresponding to the episodic steps in RL. A smaller  $RMSE$  corresponds to a more stable.

Finally, we evaluate the performance in complex environments, as depicted in Fig. 7. The assessment metrics focus on how effectively the vehicle navigates through intricate obstacles while ensuring safety and speed. We showcase the learning strength in the Sochi Circuit and the learning diversity in the Silverstone Circuit. For this evaluation, we utilized real maps and employed the metrics of "Enter and Exit Speed" and "Racing Track Lap Time" to assess the agent's performance. In summary, our results demonstrate the learning strength and diversity of the proposed algorithm in handling complex environments, and also showcase its robustness when encountering new scenarios without further training.

### 3.1. Learning performance and efficiency evaluation

The hyperparameters used in set up are listed in Table 1. Set up is aimed at verifying the efficiency of the algorithm to be applied in a real environment. Therefore, reducing the learning time is the priority. To evaluate whether learning efficiency and stability are ensure, a basic circular map is selected, and a performance comparison experiment is conducted for each RL algorithm: DQN, ODG-DQN, DDQN, and Guide-ODG-DQN. The agent model and environment used in the experiment is shown in Fig. 8.



**Figure 8.** Set up check point and arrival point. End point (red), point1 (orange), point2 (yellow), point3 (green).

**Table 1.** Hyperparameters in set up.

Hyper Parameters	
$v$	car speed $\in [0, 0.7]m/s$
$a$	action(steering angle) $\in [-1/3, 1/3]^\circ$
$\psi$	angular speed $\in [0, 0.1]m/s$
$\tau$	update target network =10,000
$\alpha$	learning Rate =0.00025
$M_{size}$	minibatch Size =64
$\gamma$	discount =0.99
$Eps$	exploration Rete=1

$$\begin{aligned}
 R &= R_g + R_v + R_\psi, \\
 R_g &\begin{cases} -200, & \text{if } crash \\ 300, & \text{if } finish \\ 100, & \text{if } checkpoint, \end{cases} \\
 R_v &= v - a * v, \\
 R_\psi &\begin{cases} 5, & \text{if } argmax_{\theta} d_L. \\ 0, & \text{otherwise} \end{cases}
 \end{aligned} \tag{16}$$

The reward function used for training is defined in Eq. (16). First, to compare the DQN and ODG-DQN algorithms for map Gazbo, we determine the number of steps in which the checkpoint is reached during training, as indicated in Table 2. In DQN, over 50% of untrained failure cases are overestimated, whereas in ODG-DQN, 10% of untrained failure cases occur, corresponding to overestimation occurrence reduced by 80%.

**Table 2.** Epochs of algorithm passing checkpoints the first time.*Fail : Overestimation*

Experiment	Algorithm(Step)							
	DQN				ODG-DQN			
	P.1	P.2	P.3	P.4	P.1	P.2	P.3	P.4
No. 1	5	10	15	20	4	7	12	18
No. 2	6	14	Fail	Fail	4	9	13	17
No. 3	7	11	16	20	4	7	12	16
No. 4	Fail	Fail	Fail	Fail	4	8	12	17
No. 5	Fail	Fail	Fail	Fail	5	8	12	18
No. 6	6	12	16	20	5	8	12	17
No. 7	5	10	16	19	4	8	13	17
No. 8	5	13	15	19	4	9	12	17
No. 9	Fail	Fail	Fail	Fail	4	8	Fail	Fail
No. 10	Fail	Fail	Fail	Fail	5	9	12	17
Average	5.75	13.3	15.6	19.6	4.3	8.9	12.2	17.1

Guide-ODG-DQN and DDQN are compared under the same conditions. Fig. 9 (a) shows the results of learning in terms of the epoch values of the safe convergence section for each algorithm implemented 10 times. As indicated in Table 3, the learning convergence rate increases by 51.7%, 89%, and 16.8%, respectively, compared with the other algorithm. Fig. 9 (b) shows that the learning is inappropriate due to overestimation in the case of DQN. In the cases of ODG-DQN, DDQN, and Guide-ODG-DQN, learning converges at approximately 500, 300, and 200 epochs, respectively the results are summarized in Table 5.

Next, to evaluate the stability of the RL results, the center line corresponding to Map Gazbo is applied as a reference. The path generated by each algorithm is shown in Fig. 9 (a). Moreover, Table 4 shows the results obtained by comparing the algorithms in terms of the RMSE, as defined in Eq. (15). The RMSE for Guide-ODG-DQN is 0.04, corresponding to the highest stability. The Guide-ODG-DQN achieves the lowest RMSE, corresponding to the highest stability, as shown in Fig. 9 (c).

**Table 3.** Decrease in the epochs of ODG-DQN.

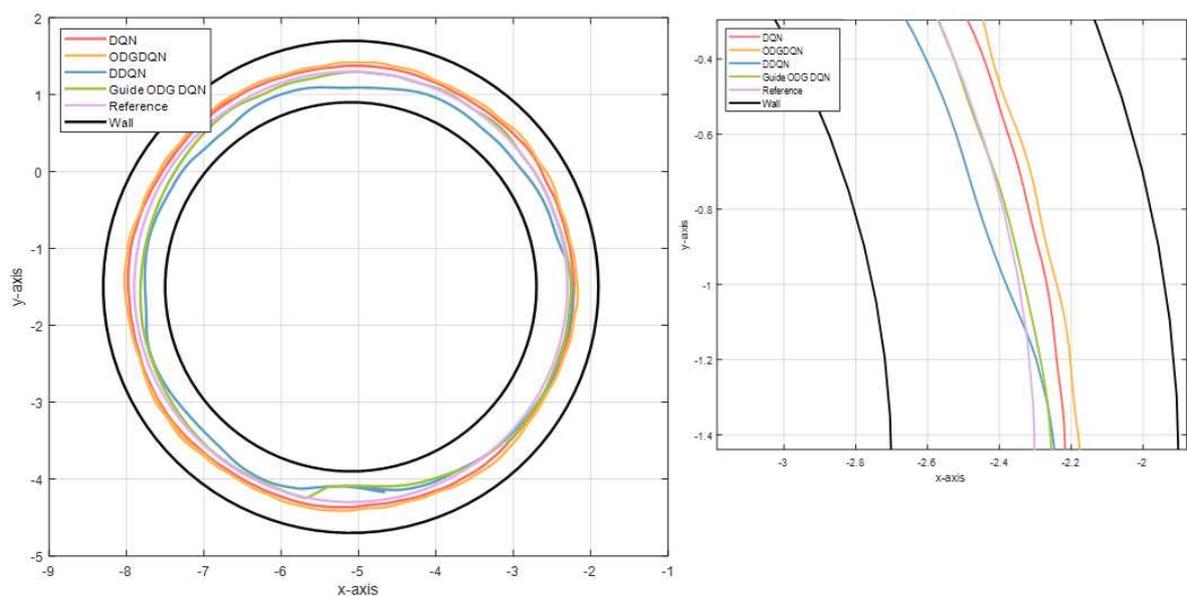
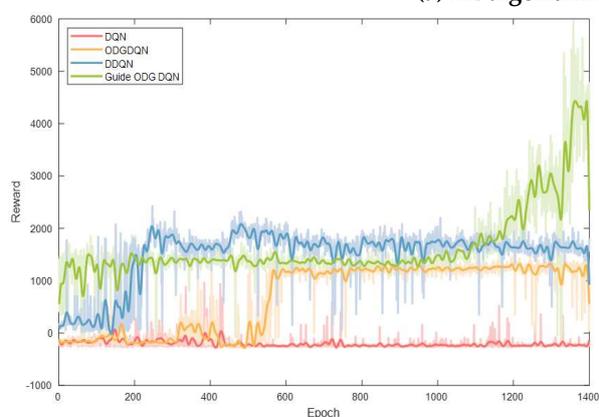
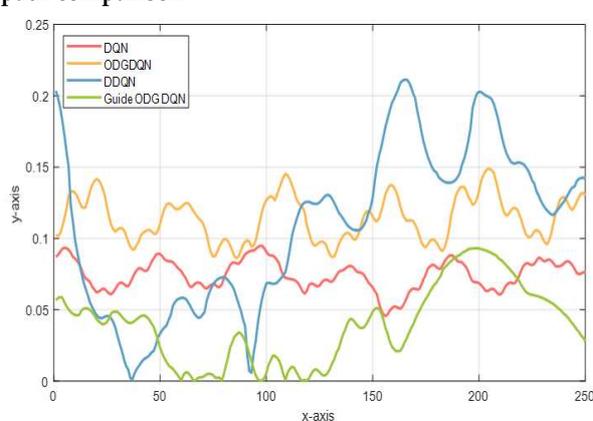
Algorithm	Decrease in the epochs
DQN → ODG-DQN	51.7%
DQN → Guide-ODG-DQN	89%
DDQN → Guide-ODG-DQN	16.8%

**Table 4.** RL RMSE.

Algorithm	RMSE
DQN	0.0745
ODG-DQN	0.1142
DDQN	0.1082
Guide-ODG-DQN	0.0395

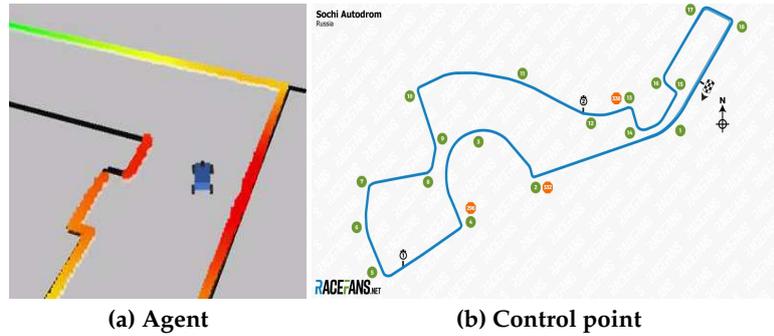
**Table 5.** Summary of normalized performance up to 10 cycles of play on track.*Fail : Overestimation*

Experiment	Algorithm(Epoch)			
	DQN	ODG DQN	DDQN	Guide ODG DQN
No. 1	1342	587	181	134
No. 2	Fail	621	175	175
No. 3	1416	576	177	121
No. 4	Fail	572	201	143
No. 5	Fail	610	182	172
No. 6	1321	593	177	144
No. 7	1422	631	188	155
No. 8	1452	579	192	177
No. 9	Fail	Fail	181	165
No. 10	Fail	668	185	143
Average Value	1391	672	184	153

**(a) RL algorithm path comparison****(b) RL reward graph****(c) RL RMS****Figure 9.** Experiment set up result

### 3.2. Results through Simulation that Mimics the Real Environmen

The hyperparameters values used in circuit are listed in Table 6. As the evaluation metric for a complex environment, shown in Fig. 10 (a), the method of learning the speed is considered instead of that for learning angles. Therefore, the angle is set to that associated with the ODG to ensure stability. The reward function used for all RL frameworks is the same as that defined in Eq. (17). Fig. 10 (b) shows the official competition map provided by F1TENTH. Using the control point specified in the actual Sochi Autodrom map, we compare the path in the winding road and hairpin curve.



**Figure 10.** Actual existing Sochi autodrom map information, officially provided by F1TENTH.

**Table 6.** Hyperparameters for the F1TENTH.

Hyperparameters	
$a$	action (car speed) $\in [0, 20] m/s$
$\Delta\psi$	ODG steering angle $\in [-12, 12]^\circ$
$max_v$	max car speed = 20 m/s
$\tau$	update target network =10,000
$\alpha$	learning Rate =0.00001
$M_{size}$	minibatch Size =128
$\gamma$	discount =0.99
$Eps$	exploration Rate=1

$$\begin{aligned}
 R &= R_\Lambda + R_W + R_g, \\
 R_\Lambda &= a - max_v - |\Delta\psi|, \\
 R_W &\in [0, 100], \\
 R_g &\begin{cases} -100, & \text{if } crash \\ 200, & \text{if } finsh \\ episode_\Lambda, & \text{if } episodestep. \end{cases}
 \end{aligned} \tag{17}$$

The agent starts at the wall of control point 1. Linear velocity graphs for ODG, Gap Follower, DDQN, Meta ODG DDQN, and are shown in Fig. 13. In this case, 100 points on the x-axis are used as control points, and 100-step linear velocity values are output on both sides based on these values.

### 3.2.1. Sochi International Street Circuit

The Sochi Autodrom previously known as the Sochi International Street Circuit and the Sochi Olympic Park Circuit, is a 5.848 km permanent race track in the settlement of Sirius next to the Black Sea resort town of Sochi in Krasnodar Krai, Russia as shown in Fig 11. In this result is demonstrated the learning strength, in the Sochi Circuit.



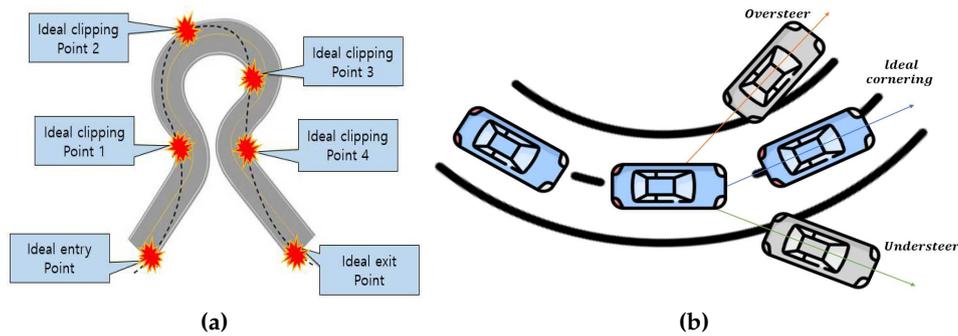
Figure 11. Circuit.

Table 7 lists the average speed for each control point for each algorithm. Table 7 shows that the ODG algorithm that prioritizes stability achieves the lowest value of 7.66, and the Meta ODG DDQN achieves the highest value of 8.58. In other words, the Meta ODG DDQN completes the Sochi Autodrom with a speed 12.01% higher than that of the ODG.

Table 7. Average speed control point.

Method	Algorithm			
	ODG	Gap Follower	DDQN	Meta ODG DDQN
No. 1	8.98	7.98	8.23	8.48
No. 2	8.10	7.79	7.93	8.69
No. 3	8.04	8.02	<b>7.78</b>	<b>8.86</b>
No. 4	7.74	7.75	8.26	8.60
No. 5	7.84	7.79	7.94	8.49
No. 6	8.41	7.95	8.59	8.66
No. 7	7.58	7.76	8.35	8.59
No. 8	7.71	7.72	8.24	<b>8.25</b>
No. 9	7.41	7.67	7.81	8.34
No. 10	7.82	7.72	8.17	8.38
No. 11	<b>9.15</b>	8.03	<b>8.66</b>	8.82
No. 12	9.08	<b>8.04</b>	8.23	8.48
No. 13	7.40	7.80	8.53	8.51
No. 14	<b>5.87</b>	7.45	8.08	8.73
No. 15	6.89	7.64	8.05	8.70
No. 16	5.44	<b>7.30</b>	7.88	8.30
No. 17	7.99	7.69	8.36	8.86
No. 18	6.34	7.42	8.12	8.61
Average speed	<b>7.66</b>	<b>7.75</b>	<b>8.17</b>	<b>8.58</b>

As shown in Fig. 12 (a), to examine the speeds of entry and exit at the control point, which are of significance in a racing game, the entry and exit speed for each algorithm are presented in Table 8. In the case of ODG, which is an algorithm that prioritizes stability, as shown in Fig. 12 (b), understeer or oversteer does not occur.[28,29] A report on racing high-performance tires [30] indicates that in this driving method, the vehicle enters at a high speed and exits at a low speed.



**Figure 12.** Corner driving (a) clipping point, (b) understeer and oversteer.

As shown in Table 9, ODG select a drive with a 13.9% speed reduction. The racing algorithm, Gap Follower, uses an out-in-out driving method with a 3.41% deceleration. However, the DDQN and Meta ODG DDQN algorithms lead to oversteer to achieve maximum speed based on the angle extracted from the ODG, which pursues stability, causing the vehicle to spin inward compared to the expected route. So DDQN and Meta ODG DDQN shows a driving method without deceleration at a control point of 1.33% and 0.46%, respectively, by drawing a path.

**Table 8.** Control point enter and exit speed.

Method	Enter and Exit speed (m/s)							
	ODG		GAP Follower		DDQN		Meta ODG DDQN	
	Enter	Exit	Enter	Exit	Enter	Exit	Enter	Exit
No.1	9.46	8.59	7.99	8.04	8.07	8.46	8.42	8.63
No.2	<b>9.28</b>	<b>6.98</b>	8.07	7.58	8.02	7.91	8.86	8.61
No.3	8.08	8.08	8.05	8.06	7.63	8.00	8.83	8.97
No.4	8.69	6.86	<b>8.07</b>	<b>7.49</b>	8.16	8.44	8.75	8.54
No.5	8.83	6.92	7.98	7.66	8.21	7.74	<b>8.54</b>	<b>8.51</b>
No.6	8.59	8.32	7.95	8.03	<b>8.61</b>	<b>8.65</b>	8.77	8.64
No.7	8.42	6.81	8.05	7.54	8.26	8.51	<b>8.83</b>	<b>8.45</b>
No.8	8.35	7.15	7.88	7.62	<b>8.63</b>	<b>7.91</b>	8.25	8.34
No.9	7.11	7.77	7.56	7.86	8.01	7.68	8.24	8.54
No.10	8.97	6.74	8.03	7.49	8.38	8.03	8.34	8.52
No.11	<b>9.14</b>	<b>9.25</b>	8.06	8.07	8.56	8.84	8.96	8.77
No.12	9.23	9.02	<b>8.07</b>	<b>8.07</b>	8.35	8.18	8.64	8.41
No.13	8.51	6.37	8.07	7.60	8.70	8.43	8.39	8.72
No.14	6.33	5.48	7.61	7.36	8.43	7.80	8.71	8.85
No.15	7.63	6.22	7.89	7.47	8.13	8.04	8.85	8.65
No.16	5.84	5.07	7.48	7.17	8.05	7.78	8.58	8.10
No.17	8.96	7.09	8.00	7.44	8.42	8.38	9.02	8.79
No.18	7.36	5.39	7.67	7.23	8.18	8.12	8.59	8.72
Average speed	<b>8.27</b>	<b>7.12</b>	<b>7.92</b>	<b>7.65</b>	<b>8.27</b>	<b>8.16</b>	<b>8.64</b>	<b>8.60</b>

**Table 9.** Enter and exit speed.

Algorithm	Speed Reduction
ODG	13.9 %
Gap Follower	3.41 %
DDQN	1.33 %
Meta ODG DDQN	0.46 %

Moreover, the lap time is compared for the map shown in Fig. 10 (b) considering two laps (based on the F1TENTH formula). Table 10 indicates that Meta ODG DDQN achieves the highest speed.

Linear velocity graphs for ODG, Gap Follower, DDQN, Meta ODG DDQN, and are shown in Fig. 13, using the control point specified in the actual Sochi circuit, we compare the path in the winding road and hairpin curve, as shown in Fig. 14.

Table 10. Racing track lap time.

Algorithm	Racing Track 2Lap Time(s)
ODG	117.09
Gap Follower	115.68
DDQN	115.41
Meta ODG DDQN(ours3)	109.85

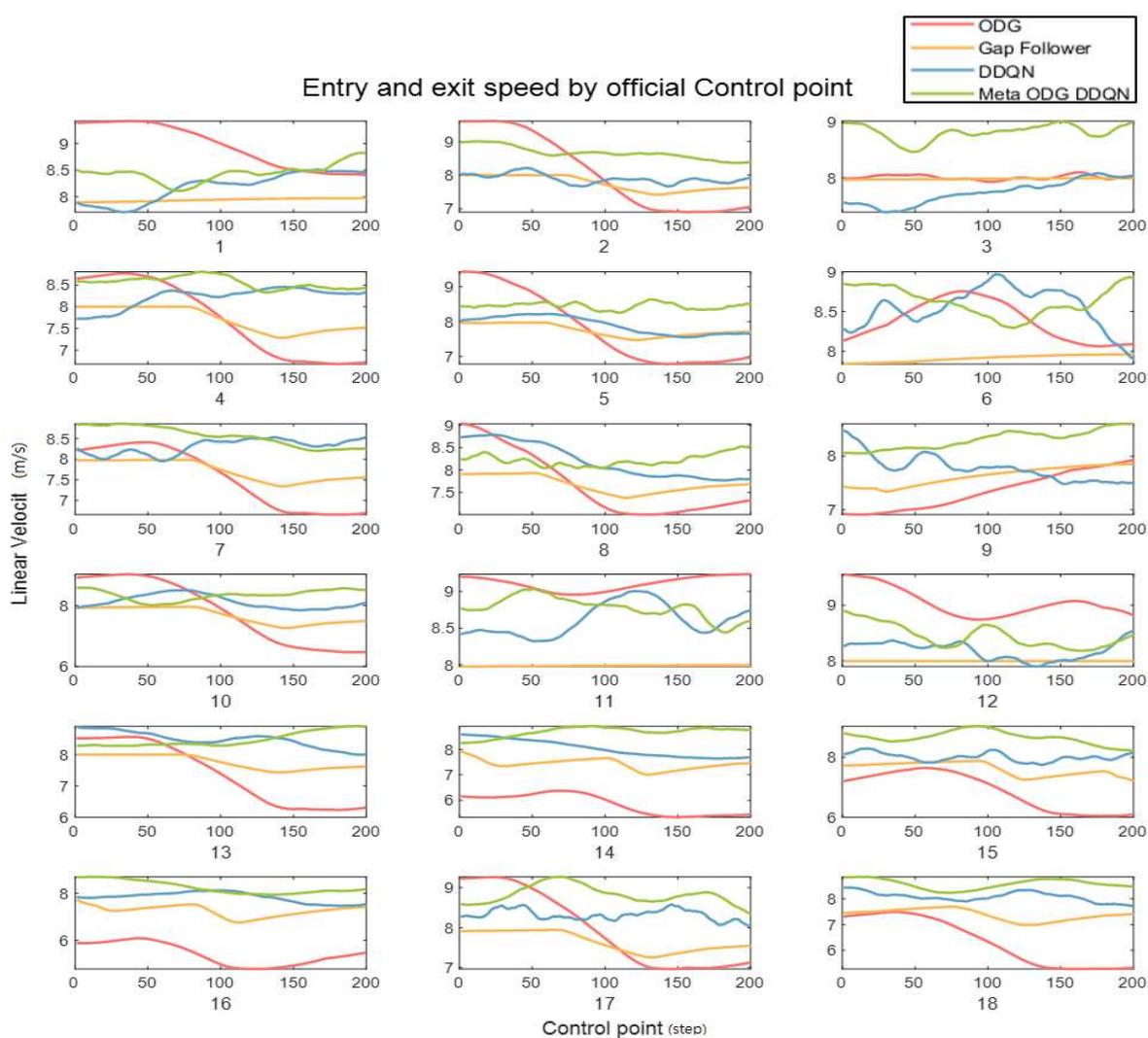
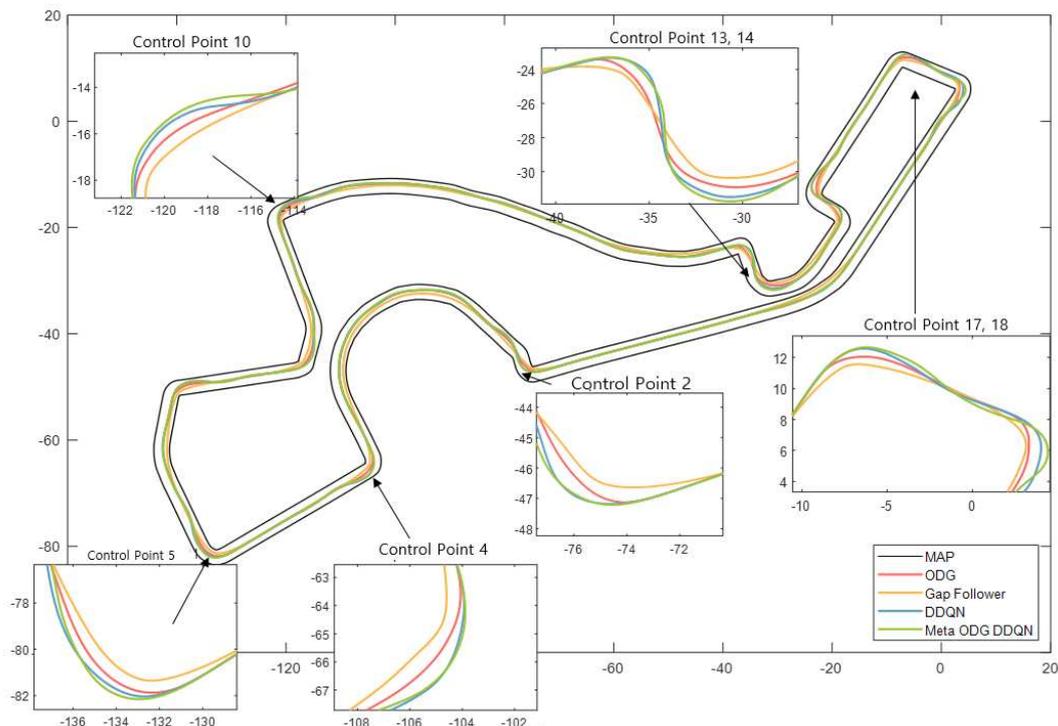


Figure 13. Control point speed.



**Figure 14.** Map of the Sochi path.

### 3.2.2. Silverstone Circuit

Silverstone Circuit is a motor racing circuit in England, near the Northamptonshire villages of Towcester, Silverstone and Whittlebury as shown in Fig 11. In this result is demonstrated the learning diversity in the Silverstone Circuit.

Using the RL model trained in Map Sochi, we conduct an experiment to determine the degree of robustness to unfamiliar and complex environments. Therefore, we use the algorithms ODG, Gap Follower, DDQN, Meta ODG DDQN. In addition, the Meta ODG DDQN algorithm is trained in a new environment. In other words, the robustness of the new environment (c) was compared based on the driving style learned in (b) shown in Fig. 7.

Table 11 presents the results for a new environment. DDQN fails; however, Meta ODG DDQN exhibits a high performance with the lowest lap time, as shows in Fig. 15. In this result is demonstrated the learning diversity in the Silverstone Circuit.

**Table 11.** Map(b) SILVERSTONE racing track lap time (two laps).

*Fail* = Crash

Algorithm	Lap Time(s)
ODG	117.39
Gap Follower	110.99
DDQN	<i>Fail</i>
Meta ODG DDQN unlearned	108.60
Meta ODG DDQN learned	108.43

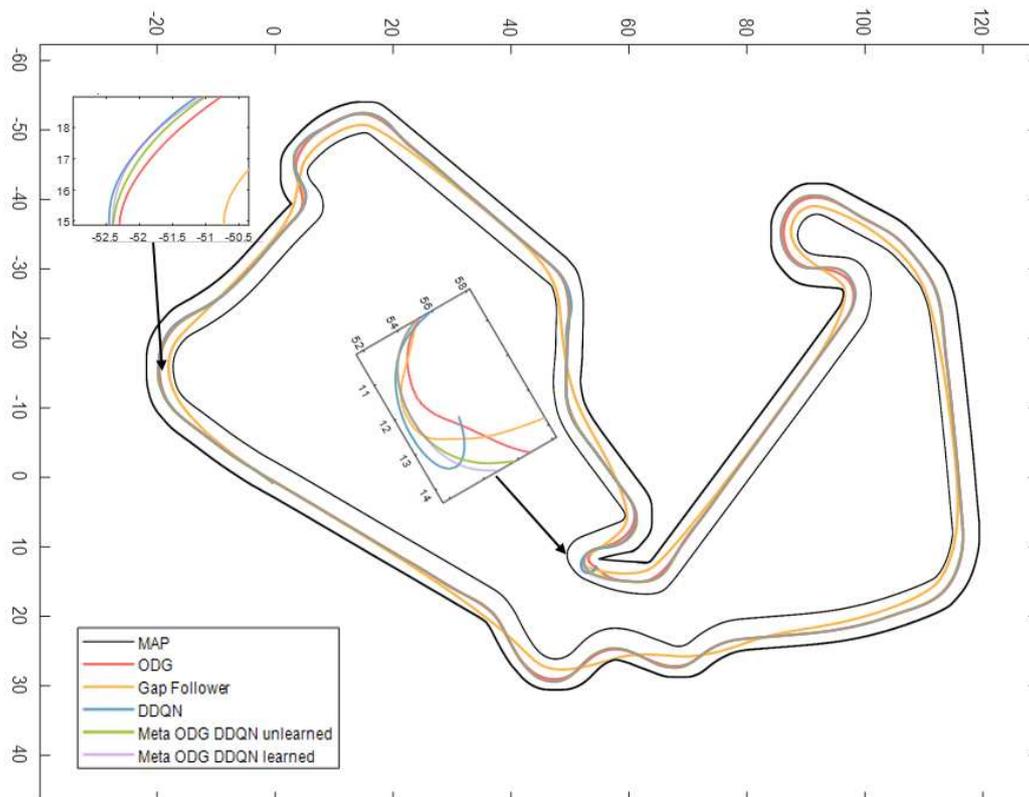


Figure 15. Map(b) SILVERSTONE path.

#### 4. Conclusions

This paper introduces a novel RL-based autonomous driving system technology that implements ODG, SAC, and meta-learning algorithms. In the autonomous driving technology, perception, decision-making, and control processes intertwine and interact. This work addresses the issues of the overvaluation phenomenon and sparse rewards problems by applying the concept of prior knowledge. Furthermore, the fusion of meta-learning-based RL yields robust results in previously untrained environments.

The proposed algorithm was tested on official F1 circuits, a racing simulation complexed dynamics environments. The results of these simulations emphasize the exceptional performance of our method, which exhibits a learning speed up to 89% faster than existing algorithms in these environments. Within the racing context, the disparity between entry and exit speeds is a mere 0.46%, indicating the smallest reduction ratio. Moreover, the average driving speed was found to be up to 12.01% higher.

The primary contributions of this paper comprise the unique combination addressing the challenges of overvaluation phenomenon and sparse rewards problems effectively in RL. Another major contribution is the demonstrated robust performance of the integrated meta-learning-based RL in previously untrained environments, thereby showcasing its adaptability and stability. Furthermore, we validated the performance of our proposed method via complex racing simulations, particularly on official F1 circuits. The results highlighted its superior performance in terms of learning efficiency, speed, stability, and adaptability.

In essence, this paper tackles the significant challenges encountered during the reinforcement learning process by introducing an algorithm that bolsters the efficiency and stability of RL. The high-fidelity simulations used in this study offer a realistic testing environment closely mirroring real-world conditions. Given these advancements, our proposed algorithm demonstrates significant potential for real-world applications, particularly in autonomous vehicles where learning efficiency and operational stability are of the importance.

**Author Contributions:** Conceptualization, J.S.H. and M.T.L.; Formal analysis, J.S.H., A.W.J. and M.T.L.; Methodology, J.S.H., K.Y.J., M.T.L. and D.S.P.; Software, J.S.H., H.G.H. and D.S.P.; Validation, M.T.L. and D.S.P.; Writing—original draft, J.S.H. and M.T.L.; Writing—review and editing, M.T.L. and D.S.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the Basic Science Research Program through the National Research Foundation of Korea(NRF) (grant no. NRF-2022R1F1A1073543).

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

**Data Availability Statement:** The code used in this paper is attached to the Github address below. <https://github.com/jang1229/Gazebo-ODG-DQN>; <https://github.com/jang1229/F1Thenth-ODGPF>

## References

1. Hoel, Carl-Johan and Wolff, Krister and Laine, Leo, "Automated speed and lane change decision making using deep reinforcement learning," *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2148-2155, 2018.
2. Qiao, Zhiqian and Muelling, Katharina and Dolan, John M and Palanisamy, Praveen and Mudalige, Priyantha, "Automatically generated curriculum based reinforcement learning for autonomous vehicles in urban environment," *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1233-1238, 2018.
3. Barreto, André and Hou, Shaobo and Borsa, Diana and Silver, David and Precup, Doina, "Fast reinforcement learning with generalized policy updates," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30079-30087, 2020.
4. Bellman, Richard, "A Markovian decision process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679-684, 1957.
5. LeCun, Yann and Bengio, Yoshua and Hinton, Geoffrey, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.
6. Peng, Baiyu and Sun, Qi and Li, Shengbo Eben and Kum, Dongsuk and Yin, Yuming and Wei, Junqing and Gu, Tianyu, "End-to-end autonomous driving through dueling double deep Q-network" *Automotive Innovation, Springer*, vol.4, pp. 328-337, 2021.
7. Yang, Yongliang and Pan, Yongping and Xu, Cheng-Zhong and Wunsch, Donald C, "Hamiltonian-driven adaptive dynamic programming with efficient experience replay" *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
8. Sutton, Richard S and Barto, Andrew G, "Reinforcement learning: An introduction," *MIT press*, 2018.
9. Gangopadhyay, Briti and Soora, Harshit and Dasgupta, Pallab, "Hierarchical program-triggered reinforcement learning agents for automated driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp.10902-10911, 2021.
10. Dayal, Aveen and Cenkeramaddi, Linga Reddy and Jha, Ajit, "Reward criteria impact on the performance of reinforcement learning agent for autonomous navigation," *Applied Soft Computing, Elsevier*, vol. 126, pp.109241, 2022.
11. Watkins, Christopher JCH and Dayan, Peter, "Q-learning," *Machine Learning*, vol.8, no. 3-4, pp. 279-292, 1992.
12. Van Hasselt, Hado and Guez, Arthur and Silver, David, "Deep reinforcement learning with double q-learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.
13. Mnih, Volodymyr and Kavukcuoglu, Koray and Silver, David and Rusu, Andrei A and Veness, Joel and Bellemare, Marc G and Graves, Alex and Riedmiller, Martin and Fidjeland, Andreas K and Ostrovski, Georg and others, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529-533, 2015.
14. Wang, Ziyu and Schaul, Tom and Hessel, Matteo and Hasselt, Hado and Lanctot, Marc and Freitas, Nando, "Dueling network architectures for deep reinforcement learning," *International Conference on Machine Learning*, 2016.
15. Burrell, Jenna, "How the machine 'thinks': Understanding opacity in machine learning algorithms," *Big Data Society*, vol. 3, no. 1, pp. 1-12, 2016.

16. Bai, Chenjia and Wang, Lingxiao and Wang, Yixin and Wang, Zhaoran and Zhao, Rui and Bai, Chenyao and Liu, Peng, "Addressing hindsight bias in multigoal reinforcement learning," *IEEE Transactions on Cybernetics*, pp. 1-14, 2021.
17. Kulkarni, Tejas D and Narasimhan, Karthik and Saeedi, Ardavan and Tenenbaum, Josh, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," *Advances in Neural Information Processing Systems*, vol. 29, pp. 3675-3683, 2016.
18. J. H. Cho and D. S. Pae and M. T. Lim and T. K. Kang, "A real-time obstacle avoidance method for autonomous vehicles using an obstacle dependent Gaussian potential field," *Journal of Advanced Transportation*, vol. 2018, pp. 1-15, 2018.
19. DS Pae and GH Kin and TK Kang and MT Lim, "Path Planning Based on Obstacle-Dependent Gaussian Model Predictive Control for Autonomous Driving," *Applied Sciences*, vol. 11, 2021.
20. J. C. Kim and D. S. Pae and M. T. Lim, "Obstacle Avoidance Path Planning based on Output Constrained Model Predictive Control," *International Journal of Control, Automation and Systems*, vol. 17, pp. 2850-2861, 2019.
21. Botvinick, Matthew and Ritter, Sam and Wang, Jane X and Kurth-Nelson, Zeb and Blundell, Charles and Hassabis, Demis, "Reinforcement learning, fast and slow," *Trends in Cognitive Sciences*, vol. 23, no. 5, pp. 408-422, 2019.
22. Sallab, Ahmad EL and Abdou, Mohammed and Perot, Etienne and Yogamani, Senthil, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70-76, 2017.
23. Korah, Thommen and Medasani, Swarup and Owechko, Yuri, "Strip histogram grid for efficient lidar segmentation from urban environments," *Computer Vision and Pattern Recognition 2011 Workshops*, pp. 74-81, 2011.
24. Ramachandran, Deepak and Amir, Eyal, "Bayesian Inverse Reinforcement Learning," *International Joint Conference on Artificial Intelligence*, vol. 7, pp. 2586-2591, 2007.
25. Haarnoja, Tuomas and Zhou, Aurick and Abbeel, Pieter and Levine, Sergey, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *International Conference on Machine Learning*, pp. 1861-1870, 2018.
26. Ueter, Niklas and Chen, Kuan-Hsun and Chen, Jian-Jia, "TProject-based CPS education: A case study of an autonomous driving student project," *IEEE Design & Test*, vol. 37, no. 6, pp. 39-46, 2020.
27. Betz, Johannes and Zheng, Hongrui and Liniger, Alexander and Rosolia, Ugo and Karle, Phillip and Behl, Madhur and Krovi, Venkat and Mangharam, Rahul, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, pp. 458-488, 2022.
28. Hosseinian Ahangarnejad, Arash and Melzi, Stefano, "Numerical analysis of the influence of an actively controlled spoiler on the handling of a sports car," *Journal of Vibration and Control*, vol. 24, no. 22, pp. 5437-5448, 2018.
29. Nguyen, Tuan Anh, "Establishing the Dynamics Model of the Vehicle Using the 4-Wheels Steering Systems," *Mathematical Modelling of Engineering Problems*, vol. 7, no. 3, 2020.
30. Paul Haney, "The Racing High-Performance Tire: Using Tires to Tune for Grip Balance (R-351)," *Society of Automotive Engineers Inc*, pp. 60-61, 2003.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.