




Article

Nondominated Policy-Guided Learning in Multi-Objective Reinforcement Learning

Man-Je Kim ¹, Hyunsoo Park ² and Chang Wook Ahn ^{1,*}

¹ AI Graduate School, Gwangju Institute of Science and Technology, Gwangju 61005, Korea; jaykim0104@gist.ac.kr

² NCSOFT, Seongnam-si 13494, Korea; hspark8312@ncsoft.com

* Correspondence: cwan@gist.ac.kr; Tel.: +82-62-715-2264

Abstract: Control intelligence is a typical field where there is a trade-off between target objectives, and researchers in this field have longed for artificial intelligence that achieves the target objectives. Multi-objective deep reinforcement learning was sufficient to satisfy this need. In particular, multi-objective deep reinforcement learning methods based on policy optimization are leading the optimization of control intelligence. However, multi-objective reinforcement learning has difficulties when finding various Pareto optimals of multi-objectives due to the greedy nature of reinforcement learning. We propose a method of policy assimilation to solve this problem. This method was applied to MO-V-MPO, one of preference-based multi-objective reinforcement learning, to increase diversity. The performance of this method has been verified through experiments in a continuous control environment.

Keywords: reinforcement learning; multi-objective optimization; real-time environment



Citation: Kim, M.-J.; Park, H.; Ahn, C.W. Nondominated Policy-Guided Learning in Multi-Objective Reinforcement Learning. *Electronics* **2022**, *11*, 1069. <https://doi.org/10.3390/electronics11071069>

Academic Editors: Jesús Ángel Román Gallego, Claudio Savaglio, María-Luisa Pérez-Delgado, Roberto García Martín and José Escudra Burrieza

Received: 17 February 2022

Accepted: 25 March 2022

Published: 28 March 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

An intelligent system refers to a type of system that, imitating the human capability to solve complex problems by means of information processing, learns to self-adapt under indeterminate and uncertain environments. The main features of an intelligent system include uncertainty management, self-adaption and self-training, and a proclivity for optimization. The technique that exploits intelligent systems to prompt machines to produce optimized results adapted for the given environment is called intelligent control. Reinforcement learning has been prevailing in the field of intelligent control [1,2]. While the conventional methods require experts to search for optimized models, reinforcement learning effectively designs a reward function to resolve the issue [3–5]. Nonetheless, in spite of its excellency at finding local optima, it suffers from the difficulty of searching for multi-objective solutions that balance trade-offs for three reasons.

First, designing a reward function for the problems with the trade-offs is highly complicated [6–8]. In general, the reward function of reinforcement learning is designed by experts with domain knowledge. Even if an expert has excellent knowledge in the field, it is almost impossible for him to design a reward function [9–11] that adapts to the growing complexity as the number of objective increases without spending a lot of resources and time. This requires a multi-objective reinforcement learning algorithm that can solve the multi-objective problem.

Second, it is difficult to find solutions in diversity due to the greedy characteristics of reinforcement learning [12–14]. Reinforcement learning agents generally continue learning in a way that is easier to obtain rewards, so in the case of multiple objectives, they are more biased toward specific objectives that tend to return better rewards. As a result, even if the reward function is distributed evenly in weight, it becomes aligned to a specific objective compared to the even Pareto distribution.

Third, reinforcement learning suffers from low stability [15–17]. Reinforcement learning collects data by interacting with the environment instead of relying on pre-built data.

Therefore, even if learning is performed with the same reward function and parameters, the agents at the end have differences in their performance. As a result, even with appropriate parameters or reward functions, if agents have been trained concurrently to find the multi-objective Pareto solutions, some agents might lag behind with respect to the original performance. As a result, it can be a difficult task to establish a smooth Pareto front.

We propose a method of policy assimilation to solve the addressed problems. This method divides the learning into four sections, evaluates the performance of each section, and replaces the policy of dominated agents with those of non-dominated agents. This method allows each agent to find the closest non-dominated agent with a lower objective preference and to copy it in order to secure stability in the agent's diversity and performance in the multi-objective program. Our method was applied to MO-V-MPO to validate its performance, which is the latest multi-objective reinforcement learning algorithm proposed by Google DeepMind, and it currently exhibits excellent performance [18]. We solved the previously addressed problems by combining the proposed method and MO-V-MPO.

2. Related Works

2.1. Reinforcement Learning

Reinforcement learning is one sort of machine learning, for which the main objective is to find optimal policy for the agent [15,17,19]. Reinforcement learning is the only technique among many other machine-learning-oriented methodologies that collects and learns data by interacting with the environment both directly and indirectly. It is thus qualified for adapting to uncertain environments where securing a sufficient amount of data is not guaranteed, which is a big blow, especially to algorithms that require a pre-built set of data. Reinforcement learning utilizes the Markov decision process (MDP), a crucial step that shapes its ability to adapt. The MDP consists of four core elements, \mathcal{S} , \mathcal{A} , \mathcal{P} , and \mathcal{R} [16,20,21]. At each timestamp t , an agent observes a set of states $s_t \in \mathcal{S}$ and commences a set of actions $a_t \in \mathcal{A}$ that determines the reward $r_t \sim \mathcal{R}(s_t, a_t)$ and the subsequent state $s_{(t+1)} \sim \mathcal{P}(s_t, a_t)$. An MDP-defined reinforcement learning algorithm, unlike supervised learning algorithms, is not explicitly corrected for its undesirable behaviors; its behavioral tendency is only indirectly influenced by its reward function.

Reinforcement learning performs training via two-fold steps of exploration and exploitation [22,23]. During exploration in the early stage of training, the agent attempts actions randomly. As mentioned earlier, since the data must be collected by the algorithm itself, which action produces high expected rewards is unknown at the beginning. Therefore, the random actions by the agent are intended to provide diversity to the policy convergence, contributing to discovering the global optima. During exploitation, on the other hand, it induces the agent to choose actions with the highest expected rewards. It is mostly conducted in the later stage, when the data have been secured sufficiently, in order to help the agent suit the optimal policy convergence. Reinforcement learning, overall, is a proper combination of exploration and exploitation [15,24].

2.2. Multi-Objective Reinforcement Learning (MORL)

MORL is a technique to handle multi-objective (MO) problems using reinforcement learning (RL). Since many real-world problems have multiple goals to achieve and constraints to maintain, it is useful to regard them as MO optimization problems, but in the field of RL, simple reward scalarization (RS) is dominant. MORL is either single-policy or multiple-policy-based. The single-policy version finds the optimal policy in a given RS, in general by scalarizing every reward in the form of a weighted sum. On the other hand, the multiple-policy version is a method to approximate the Pareto front to find diverse policy sets. While a simple RS technique tends to be useful in the simple-policy MORL, it is often difficult to find diverse policies to sufficiently approximate the Pareto front with it. Carefully adjusting each reward's weight does not contribute significantly to searching various policies. However, the recently published studies on multi-objective

MPO (MO-MPO) and MO-V-MPO show that these techniques overcome the shortcomings of RS and can thus be trained with diverse policies [18].

2.3. Maximum a Posteriori Policy Optimization (MPO) and V-MPO

MPO [25] is one of the recently released off-policy RL algorithms of the “RL as inference” family. The purpose of the general RL algorithms is to find out which action to choose to maximize future rewards, but the MPO aims to change its perspective and estimate which action is likely to be selected when the future reward is maximized. MPO has the advantage of using expectation-maximization (EM) to increase sample efficiency and lower the variation of expected returns. E-step estimates the distribution of the action, and M-step fits the policy.

V-MPO [26] is the on-policy variant of MPO, which was originally an off-policy RL algorithm. It uses a state-value function instead of an action-value function that uses MPO. Due to the low variance of the expected return, it shows the relatively stable performance compared to the precedent algorithms, such as IMPALA [27].

$$\mathcal{L}_\pi(\theta) = - \sum_{s,a \sim \tilde{D}} \psi(s,a) \log \pi_\theta(a|s), \quad \psi(s,a) = \frac{\exp\left(\frac{A^{target}(s,a)}{\eta}\right)}{\sum_{s,a \sim \tilde{D}} \exp\left(\frac{A^{target}(s,a)}{\eta}\right)}, \quad (1)$$

Equation (1) shows the policy loss of V-MPO. It is similar to the conventional policy loss of policy gradient algorithms such as PPO and IMPALA, but there are two differences. First, it uses only the top 50% of the dataset (\tilde{D}) in terms of the advantage. Second, it augments the advantage (A^{target}) by temperature η :

$$\mathcal{L}_\eta(\eta) = \eta \epsilon_{eta} + \eta \log \left[\frac{1}{|\tilde{D}|} \sum_{s,a \sim \tilde{D}} \exp\left(\frac{A^{target}(s,a)}{\eta}\right) \right], \quad (2)$$

η is a trainable parameter, and Equation (2) shows its loss. η augments A^{target} , and ϵ_η regulates A^{target} to maintain a certain range of A^{target} . Even in the cases of extremely low A^{target} , often because of rare rewards, regulation of η adjusts the value of A^{target} with respect to ϵ_η to a sufficient degree to improve the policy. It is comparable to the advantage standardization (normalization by mean and SD), mostly used in the algorithms such as PPO [28], but the fact that it does not presuppose a certain distribution makes it more flexible.

2.4. Multi-Objective MPO (MO-MPO) and MO-V-MPO

MO-MPO and MO-V-MPO are the variants of MPO and V-MPO to apply to multi-objective problems. (V-)MPO trains a parameter η to augment the Q-value for MPO (advantages for V-MPO), and use ϵ_η to regulate it. If ϵ_η is increased, the augmented values will be increased, and vice versa. MO-(V-)MPO extends this concept to multiple objectives by letting each objective contain its η and corresponding ϵ_η . In this setting, each ϵ_η acts as a preference. We can assign a higher ϵ_η for an important objective and a low ϵ_η for a less important objective or penalty.

In usual MO environments, simple reward scalarization (RS) can only consider the amplitude of reward, but it is difficult to consider distributions or frequency of Q (action-value) and A (advantage). Since it is difficult to consider the changes of reward distributions throughout the learning progress, it is difficult to train for diverse policies using RS. On the other hand, MO-(V-)MPO can adjust η with regard to the training environment, allowing the augmented Q and A to adjust themselves according to each objective's ϵ_η . Therefore, it can train more diverse policies than RS.

3. Proposed Method

The main idea of proposed policy assimilation is to replace the policy of an agent with dominated performance with that of an agent with non-dominated performance. To this end, our method proceeds in three major steps sequentially (Figure 1, Algorithm 1).

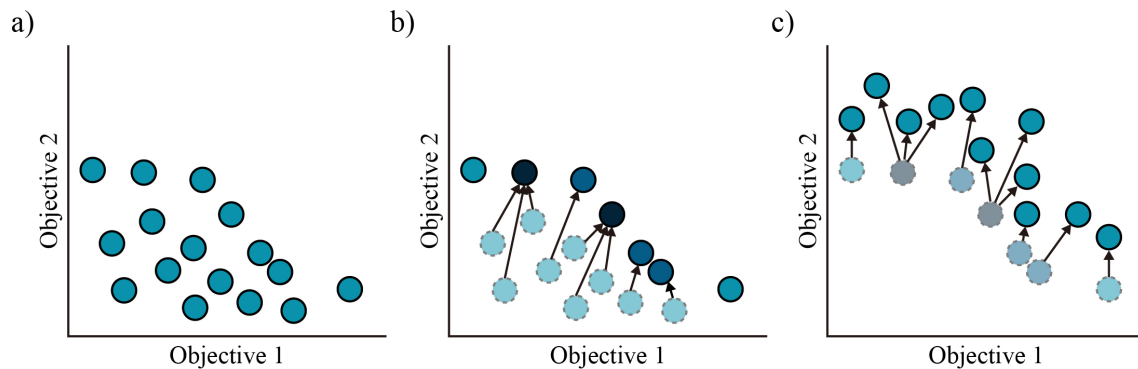


Figure 1. Graphical representation of the main processes in policy assimilation. (a) Policy Evaluation. (b) Policy Transfer. (c) Retraining.

Algorithm 1 Policy assimilation

Require: Set of Agents Π

Require: Maximum training step t_{max}

Require: Policy transfer rate $\alpha = 0.25$

1: **for** $t \leftarrow 1, t_{max}$ **do**

2: **if** $t \bmod \alpha \cdot t_{max} == 0$ **then**

3: POLICYEVALUATION(Π_t)

4: $\Pi_{t+1} \leftarrow$ POLICYTRANSFER(Π_t)

5: **else**

6: $\Pi_{t+1} \leftarrow$ PREFERENCE-BASED MULTI-OBJECTIVE RL(Π_t) \triangleright e.g., MO-V-MPO

7: **end if**

8: **end for**

The first is policy evaluation, which is performed upon all the agents every quarter of the total learning time. Based on its result, it is determined whether each agent has been dominated.

The second is policy transfer, which is applied only to dominated agents. If the agent is dominated, it will comply with the policy of the nearest non-dominated agent. The distance is determined by the differences in preference of non-dominated agents. For example, assuming that the preference is two dimensional, suppose there is a dominated agent with the pair of preferences [10, 90] and non-dominated agents with [20, 80], [70, 30], and [50, 50], respectively. Then, the dominated agent will abide by the policy of the agent possessing the preference [20, 80], which has the smallest difference of [10, 90]. If the differences in pairs of preferences are the same (e.g., [5, 95], [15, 85]), the preference moves closer to the center, which is [50, 50]. If all the agents are non-dominated, no agent's policy will change. These procedures can raise the stability of reinforcement learning performance in our method. Even if a particular agent has shown incompetent results at the beginning of learning, it can partially improve. It is worth mentioning that since MO-V-MPO is one sort of reinforcement learning, it can also be trained toward a somewhat biased direction that satisfies certain objectives. However, the treatment of moving towards the center under the situation of identical preference differences will attenuate this tendency (Algorithm 2).

Algorithm 2 Policy transfer**Require:** Set of Agents Π **Require:** Center Preference $c \leftarrow [50, 50]$ **Require:** Metric Function (e.g., Euclidean distance) DISTANCE

```

1:  $P, Q \leftarrow \text{FAST NON-DOMINATED SORTING}(\Pi)$            ▷ P: non-dominated agents, Q:
   dominated agents
2: for  $q \in Q$  do
3:    $d_{min} \leftarrow \infty$ 
4:   for  $p \in P$  do
5:      $d \leftarrow \text{DISTANCE}(q.preference, p.preference)$ 
6:     if  $d_{min} > d$  then
7:        $d_{min} \leftarrow d$ 
8:        $p_o \leftarrow p$ 
9:     else if  $d_{min} == d$  then
10:       $d_1 \leftarrow \text{DISTANCE}(c, p.preference)$ 
11:       $d_2 \leftarrow \text{DISTANCE}(c, p_o.preference)$ 
12:      if  $d_1 < d_2$  then
13:         $p_o \leftarrow p$ 
14:      end if
15:    end if
16:  end for
17:   $q.policy \leftarrow p_o.policy$ 
18: end for
19: return  $\Pi_{new} \leftarrow P \cup Q$ 

```

Lastly, the retraining step is the process of training again after the policy is transferred. Each agent maintains the existing preference regardless of whether or not the policy has been transferred and repeats the training. The learning direction according to the preference is then maintained, so the agent performs learning toward each objective, resulting in more stable performance, while increasing the agent's diversity.

4. Experiment and Results

We used a continuous control environment to prove the performance of the proposed method: Lunar Lander. The main goal of an agent in this environment is to land quickly and safely on the destination point of the moon. If the agent is alive until the end of the episode, it receives 100 points, and if it perishes, it receives -100 . When the agent arrives at the destination with the speed of zero, it receives an additional 100 to 140 points, the difference being that an extra 10 points are given per leg that has made a contact with the ground. In addition, a penalty of -0.3 points per frame is given if the center engine runs at 50% of the maximum thrust, -0.03 points for the left and right engines. The engines are only active when they run at above 50% of the maximum thrust, and the penalty points increase exponentially with respect to the engine thrust until 100%, where they double. The hyper parameters of MO-V-MPO for our experiments are described in Table 1.

Table 1. Hyper parameters for MO-V-MPO.

Hyper Parameter	Contents
Optimizer	Adam
Learning rate	10^{-4}
batch size	256
λ	0.99
γ	0.99
Initial η	0.01

The configuration of Lunar Lander promotes the strategy of starting with a slow initial speed and arriving at the destination as safely as possible in order to earn the highest score possible. However, we added the time penalty to convert it to a multi-objective problem. The time penalty point was -0.1 per step, and the maximum number of steps was 300. Therefore, the agent had to consider energy and time efficiencies at the same time. We set the preference to change from $[0, 1]$ to $[1, 0]$ with steps of 0.02 and rounded up and converted to whole numbers all the resulting rewards to manifest the outcome more clearly. Since the environment is a continuous control problem, we formed each group of proximal values into single value to determine the sectional densities. This processing allowed the result from each agent to be transcribed to a specific discrete value to demonstrate how well the agents were distributed (Figure 2).

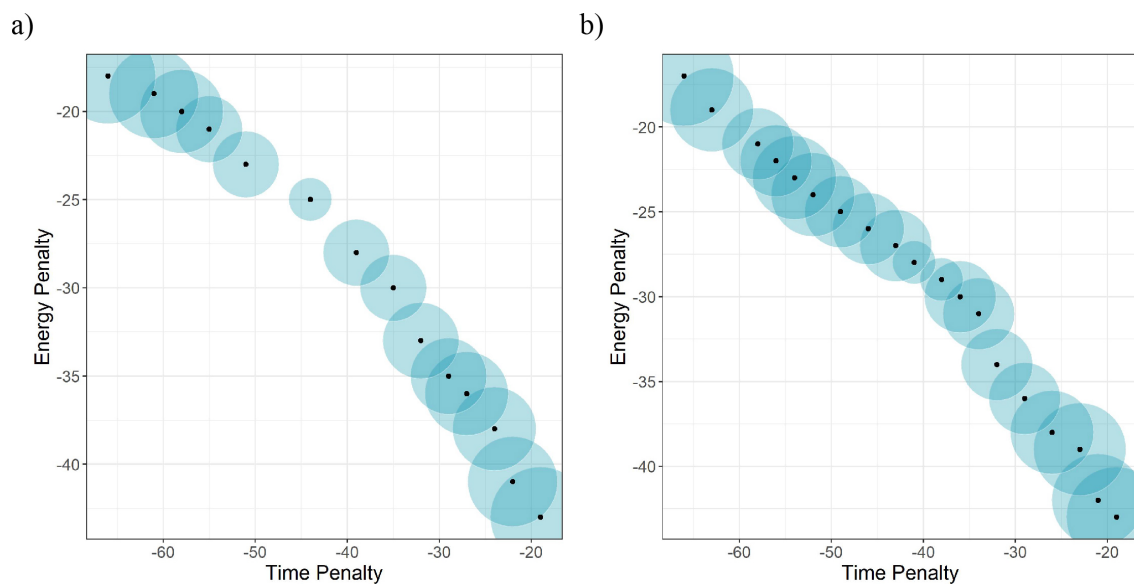


Figure 2. The black dots form the Pareto front we located throughout the experiment, and the blue circles portray by their sizes the numbers of solutions in proximity, indicating that larger circles contain more solutions. (a) is the conventional MO-V-MPO distribution, and (b) is the proposed method's distribution.

Figure 2 shows that our method had more evenly distributed results than the conventional methods. In particular, it prevailed over the precedent reinforcement learning-based multi-purpose algorithms in securing diversity. This difference is particularly observable in the range of -50 to -40 for the time penalty and -30 to -25 for the energy penalty. Similarly to other reinforcement learning algorithms, the training was not processed well in the sectors where the preferences of the two objectives overlapped, although MO-V-MPO found a few solutions in that case. While the solutions biased toward energy and time appeared mostly for both algorithms, ours produced more evenly distributed results. We repeated the same experiment 10, 50, and 100 times for the sake of fairness and compared

the numbers of sets that reached the time penalty in the range of -50 to -40 and the energy penalty in the range of -30 to -25 . The overall results are in Table 2.

Table 2. Comparison on the number of agents that reached the contact point.

	MO-V-MPO	MO-V-MPO with Policy Assimilation
10 times	1	1
50 times	1	4
100 times	2	7

5. Conclusions

In this paper, we proposed a method of policy assimilation, and showed that it resolves the problems of biased learning for objectives in multi-objective reinforcement learning and trains appropriately for various objectives. Our method is exceptional, since, unlike most previous reinforcement learning studies that focus on improving the overall performance, it concentrates mainly on the diversity of agents. The performance of our proposed method has been validated via single, limited environment, Lunar Lander. From this fact we admit that the experiment did not consider diverse environments. However, we would also like to emphasize that there were virtually no other continuous control problems we could find that well manifest the contrast between energy and time. Furthermore, the environment we conducted the experiment in can be said to be an abstract version of the real-world trade-offs in solving problems, such as heating and cooling systems, which are regarded as the most challenging control problems. Therefore, we anticipate that our study can be used as a cornerstone for future challenges in real-world problems with complex energy and time trade-offs. Lastly, the provision of diversity in our study is eligible for providing various policies adjusted for numerous users of reinforcement learning AIs to solve the trade-off problems, thereby better reflecting their intended objectives.

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

Funding: This work was supported by the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2021R1A2C3013687 and NRF-2020R1A6A3A13055636).

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

References

1. Nguyen, T.T.; Nguyen, N.D.; Nahavandi, S. Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications. *IEEE Trans. Cybern.* **2020**, *50*, 3826–3839. [[CrossRef](#)] [[PubMed](#)]
2. Wen, Y.; Si, J.; Brandt, A.; Gao, X.; Huang, H.H. Online Reinforcement Learning Control for the Personalization of a Robotic Knee Prosthesis. *IEEE Trans. Cybern.* **2020**, *50*, 2346–2356. [[CrossRef](#)] [[PubMed](#)]
3. Busoniu, L.; Babuska, R.; De Schutter, B. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Trans. Cybern.* **2008**, *38*, 156–172. [[CrossRef](#)]
4. Nguyen, T.T.; Nguyen, N.D.; Vamplew, P.; Nahavandi, S.; Dazeley, R.; Lim, C.P. A multi-objective deep reinforcement learning framework. *Eng. Appl. Artif. Intell.* **2020**, *96*, 103915. [[CrossRef](#)]
5. Van Seijen, H.; Fatemi, M.; Romoff, J.; Laroche, R.; Barnes, T.; Tsang, J. Hybrid Reward Architecture for Reinforcement Learning. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
6. Van Moffaert, K.; Drugan, M.M.; Nowé, A. Scalarized multi-objective reinforcement learning: Novel design techniques. In Proceedings of the 2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), Singapore, 16–19 April 2013; pp. 191–199.
7. Liu, C.; Xu, X.; Hu, D. Multiobjective Reinforcement Learning: A Comprehensive Overview. *IEEE Trans. Syst. Man Cybern. Syst.* **2015**, *45*, 385–398.

8. Parisi, S.; Pirota, M.; Restelli, M. Multi-objective reinforcement learning through continuous Pareto manifold approximation. *J. Artif. Intell. Res.* **2016**, *57*, 187–227. [[CrossRef](#)]
9. Ng, A.Y.; Harada, D.; Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In Proceedings of the Sixteenth International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, USA, 14–16 June 1999; pp. 278–287.
10. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Pearson: Boston, MA, USA, 2020.
11. Lee, D.; Seo, H.; Jung, M.W. Neural basis of reinforcement learning and decision making. *Annu. Rev. Neurosci.* **2012**, *35*, 287–308. [[CrossRef](#)] [[PubMed](#)]
12. Khamis, M.A.; Gomaa, W. Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. *Eng. Appl. Artif. Intell.* **2014**, *29*, 134–151. [[CrossRef](#)]
13. Mossalam, H.; Assael, Y.M.; Roijers, D.M.; Whiteson, S. Multi-objective deep reinforcement learning. *arXiv* **2016**, arXiv:1610.02707.
14. Ruiz-Montiel, M.; Mandow, L.; de la Cruz, J.L.P. A temporal difference method for multi-objective reinforcement learning. *Neurocomputing* **2017**, *263*, 15–25. [[CrossRef](#)]
15. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 1998.
16. Kim, M.J.; Kim, J.S.; Kim, S.J.; Kim, M.; Ahn, C.W. Genetic State-Grouping Algorithm for Deep Reinforcement Learning. *Expert Syst. Appl.* **2020**, *161*, 113695. [[CrossRef](#)]
17. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**, arXiv:1312.5602.
18. Abdolmaleki, A.; Huang, S.; Hasenclever, L.; Neunert, M.; Song, F.; Zambelli, M.; Martins, M.; Heess, N.; Hadsell, R.; Riedmiller, M. A distributional view on multi-objective policy optimization. In *International Conference on Machine Learning*; PMLR: Stockholm, Sweden, 2020; pp. 11–22.
19. Hasselt, H.V.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 2094–2100.
20. Kapturowski, S.; Ostrovski, G.; Dabney, W.; Quan, J.; Munos, R. Recurrent Experience Replay in Distributed Reinforcement Learning. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
21. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.P.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous Methods for Deep Reinforcement Learning. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1928–1937.
22. 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 International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.
23. Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; Abbeel, P. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In Proceedings of the International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016. [[CrossRef](#)]
24. Wang, Z.; Schaul, T.; Hessel, M.; van Hasselt, H.; Lanctot, M.; de Freitas, N. Dueling Network Architectures for Deep Reinforcement Learning. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016.
25. Abdolmaleki, A.; Springenberg, J.T.; Tassa, Y.; Munos, R.; Heess, N.; Riedmiller, M. Maximum a posteriori policy optimisation. *arXiv* **2018**, arXiv:1806.06920.
26. Song, H.F.; Abdolmaleki, A.; Springenberg, J.T.; Clark, A.; Soyer, H.; Rae, J.W.; Noury, S.; Ahuja, A.; Liu, S.; Tirumala, D.; et al. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. *arXiv* **2019**, arXiv:1909.12238.
27. Espeholt, L.; Soyer, H.; Munos, R.; Simonyan, K.; Mnih, V.; Ward, T.; Doron, Y.; Firoiu, V.; Harley, T.; Dunning, I.; et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*; PMLR: Stockholm, Sweden, 2018; pp. 1407–1416.
28. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.