Combining Evolution and Deep Reinforcement Learning for Policy Search: a Survey

OLIVIER SIGAUD, Sorbonne Université, CNRS, Institut des Systèmes Intelligents et de Robotique, ISIR F-75005 Paris, France

Deep neuroevolution and deep Reinforcement Learning have received a lot of attention in the last years. Some works have compared them, highlighting their pros and cons, but an emerging trend combines them so as to benefit from the best of both worlds. In this paper, we provide a survey of this emerging trend by organizing the literature into related groups of works and casting all the existing combinations in each group into a generic framework. We systematically cover all easily available papers irrespective of their publication status, focusing on the combination mechanisms rather than on the experimental results. In total, we cover 45 algorithms more recent than 2017. We hope this effort will favor the growth of the domain by facilitating the understanding of the relationships between the methods, leading to deeper analyses, outlining missing useful comparisons and suggesting new combinations of mechanisms.

CCS Concepts: • **Computing methodologies** → **Reinforcement learning**.

Additional Key Words and Phrases: Evolutionary algorithms

1 INTRODUCTION

The idea that the extraordinary adaptive capabilities of living species results from a combination of evolutionary mechanisms acting at the level of a population and learning mechanisms acting at the level of individuals is ancient in life sciences [91] and has inspired early work in Artificial Intelligence (AI) research [31]. This early starting point has led to the independent growth of two bodies of formal frameworks, evolutionary methods and reinforcement learning (RL). The early history of the evolutionary side is well covered in [2] and from the RL side in [95]. Despite these independent developments, research dedicated to the combination has remained active, in particular around Learning Classifier Systems [43, 89] and studies of the Baldwin effect [105]. A broader perspective and survey on all the evolutionary and RL combinations anterior to the advent of the so called "deep learning" methods using large neural networks can be found in [19].

In this paper, we propose a survey of a renewed approach to this combination that builds on the unprecedented progress made possible in evolutionary and deep RL methods by the growth of computational power and the availability of efficient libraries to use deep neural networks. As this survey shows, the topic is rapidly gaining popularity with a wide variety of approaches and even emerging libraries dedicated to their implementation [98]. Thus we believe it is the right time for laying solid foundations to this growing field, by listing the approaches and providing a unified view that encompasses them. There are recent surveys about the comparison of evolutionary and RL methods [59, 77] which mention the emergence of some of these combinations. With respect to these surveys, ours is strictly focused on the combinations and attempts to provide a list of relevant papers as exhaustive as possible at the time of its publication, irrespective of their publication status.

Author's address: Olivier Sigaud, olivier.sigaud@isir.upmc.fr, Sorbonne Université, CNRS, Institut des Systèmes Intelligents et de Robotique, ISIR F-75005 Paris, France.

2688-3007/2022/10-ART $15.00
https://doi.org/10.1145/3569096

This survey is organized into groups of algorithms using the evolutionary part for the same purpose. In Section 2, we first review algorithms where evolution is looking for efficient policies, that is combining deep neuroevolution and deep RL. We then cover in Section 3 algorithms where evolution directly looks for efficient actions in a given state rather than for policies. In Section 4, we cover the combination of deep RL algorithms with diversity seeking methods. Finally, in Section 5, we cover various other uses of evolutionary methods, such as optimizing hyperparameters or the system's morphology. To keep the survey as short as possible, we consider that the reader is familiar with evolutionary and RL methods in the context of policy search, and has a good understanding of their respective advantages and weaknesses. We refer the reader to [88] for an introduction of the methods and to surveys about comparisons to know more about their pros and cons [59, 77].

2 EVOLUTION OF POLICIES FOR PERFORMANCE

The methods in our first family combine a deep neuroevolution loop and a deep RL loop. Figure 1 provides a generic template to illustrate such combinations. The methods optimize the performance of a policy by searching in the space of policy parameters. The central question left open by the template is **how both loops interact with each other**. Note that this template is not much adapted to account for works where the combination is purely sequential, such as [40] or the Goal Exploration Process Policy Gradient (GEP-PG) algorithm [12]. Besides, to avoid any confusion with the multi-agent setting, note that agents are interacting in isolation with their own copy of the environment and cannot interact with each other.

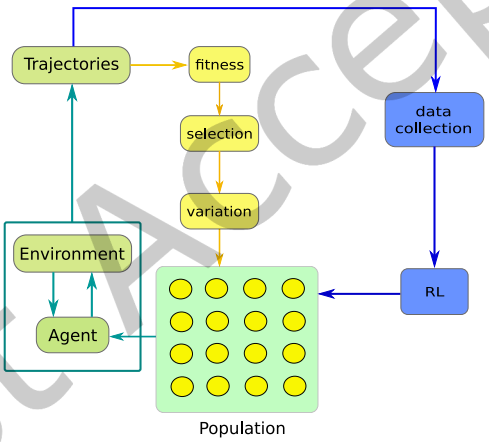

Fig. 1. The general template of algorithms combining deep neuroevolution and deep RL. A population of agents interact with an environment, and produce trajectories composed of states, actions and rewards. From the left-hand side, an evolutionary loop selects and evolves these agents based on their fitness, which is computed holistically over trajectories. From the right-hand side, a deep RL loop improves one or several agents using a gradient computed over the elementary steps of trajectories stored into a replay buffer.

The main motivation for combining evolution and deep RL is the improved performance that may result from the combination. For instance, through simple experiments with simple fitness landscapes and simplified versions of the components, combining evolution and RL can be shown to work better than using either of the two in isolation [100]. Why is this so? One of the explanations is the following. A weakness of policy gradient methods at the heart of deep RL is that they compute an estimate of the true gradient based on a limited set of samples. **This gradient can be quite wrong due to the high variance of the estimation, but it is**

Table 1. Combinations evolving policies for performance. The table states whether the algorithms in the rows use the mechanisms in the columns. The colors are as follows. In the column about other combination mechanisms (+ Comb. Mech.): Critic gradient addition • (green), Population from Actor ♠ (blue), None **x** (red). In all other columns: • (green): yes, **x** (red): no. In BNET, BBNE stands for Behavior-Based NeuroEvolution and CGP stands for Cartesian Genetic Programming [62]. The different GA labels stand for various genetic algorithms, we do not go into the details.

Prop. / Algo.	RL algo.	Evo. algo.	Actor Injec.	+ Comb. Mech.	Surr. Fitness	Soft Update	Buffer Filt.
ERL [38]	DDPG	GA	•	x	x	x	x
CERL [37]	TD3	GA	•	x	x	x	x
PDERL [4]	TD3	GA	•	x	x	x	x
ESAC [94]	SAC	ES	•	x	x	x	x
FIDI-RL [85]	DDPG	ARS	•	x	x	x	x
X-DDPG [20]	DDPG	GA	•	x	x	x	x
CEM-RL [75]	TD3	CEM	x	•	x	x	x
CEM-ACER [96]	ACER	CEM	x	•	x	x	x
SERL [102]	DDPG	GA	•	x	•	x	x
SPDERL [102]	TD3	GA	•	x	•	x	x
PGPS [41]	TD3	CEM	•	x	•	•	x
BNET [92]	BBNE	CGP	•	x	•	x	x
CSPC [108]	SAC + PPO	CEM	•	x	x	x	•
SUPE-RL [61]	RAINBOW or PPO	GA	•	♠	x	•	x
G2AC [5]	A2C	GA	x	♠	x	x	x
G2PPO [5]	PPO	GA	x	♠	x	x	x

applied blindly to the current policy without checking that this actually improves performance. By contrast, **variation-selection methods at the heart of evolutionary methods evaluate all the policies they generate and remove the poorly performing ones**. Thus a first good reason to combine policy gradient and variation-selection methods is that the latter may remove policies that have been deteriorated by the gradient step. Below we list different approaches building on this idea. This perspective is the one that gives rise to the largest list of combinations. We further split this list into several groups of works in the following sections.

2.1 Deep RL actor injection

One of the main algorithms at the origin of the renewal of combining evolution and RL is Evolutionary Reinforcement Learning (ERL) [38], see Figure 2a. It was published simultaneously with the Genetic-Gated RL algorithms G2AC and G2PPO [5] but its impact was much greater. Its combination mechanism **injects the RL actor into the evolutionary population**.

The ERL algorithm was soon followed by Collaborative Evolutionary Reinforcement Learning (CERL) [37] which extends ERL from RL to *distributed* RL where several agents learn in parallel, and all these agents are injected into the evolutionary population. The main weakness of ERL and CERL is their reliance on a genetic algorithm which applies a standard n-point-based crossover and a Gaussian weight mutation operator to a direct encoding of

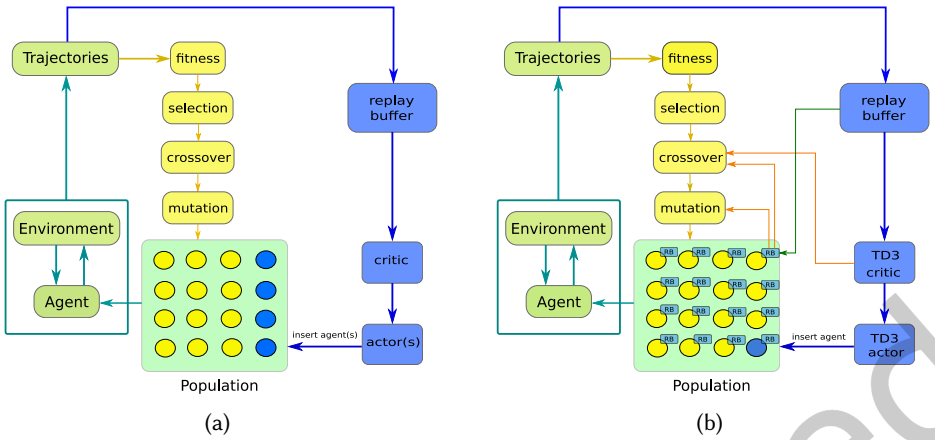

Fig. 2. The template architecture for ERL, ESAC, FIDI-RL and CERL (a) and the PDERL architecture (b). In ERL, an actor learned by DDPG is periodically injected into the population and submitted to evolutionary selection. If DDPG performs better than the GA, this will accelerate the evolutionary process. Otherwise the DDPG agent is just ignored. In ESAC, DDPG is replaced by SAC and in FIDI-RL, the GA is replaced by ARS [60]. In CERL, the DDPG agent is replaced by a set of TD3 actors sharing the same replay buffer, but each using a different discount factor. Again, those of such actors that perform better than the rest of the population are kept and enhance the evolutionary process, whereas the rest are discarded by evolutionary selection. In PDERL, the genetic operators of ERL are replaced by operators using local replay buffers so as to better leverage the step-based experience of each agent.

the neural network architecture as a simple vector of parameters. This approach is known to require tedious hyperparameter tuning and generally perform worse than evolution strategies which are also mathematically more founded [9, 79]. In particular, the genetic operators used in ERL and CERL based on a direct encoding have been shown to induce a risk of catastrophic forgetting of the behavior of efficient individuals.

The Proximal Distilled Evolutionary Reinforcement Learning (PDERL) algorithm [4], see Figure 2b, builds on this criticism and proposes two alternative evolution operators. Instead of standard crossover, all agents carry their own replay buffer and crossover selects the best experience in both parents to fill the buffer of the offspring, before applying behavioral cloning to get a new policy that behaves in accordance with the data in the buffer. This operator is inspired by the work of [23]. For mutation, they take as is the improved operator proposed in [45], which can be seen as applying a Gauss-Newton method to perform the policy gradient step [72].

Another follow-up of ERL is the Evolutionary Soft Actor Critic (ESAC) algorithm [94]. It uses the Soft Actor Critic (SAC) algorithm [27] instead of Deep Deterministic Policy Gradient (DDPG) [49] and a modified evolution strategy instead of a genetic algorithm, but the architecture follows the same template. Similarly, the Finite Difference RL (FIDI-RL) algorithm [85] combines DDPG with Augmented Random Search (ARS), a finite difference algorithm which can be seen as a simplified version of evolution strategies [60]. FIDI-RL uses the ERL architecture as is. The method is shown to outperform ARS alone and DDPG alone, but neither ESAC nor FIDI-RL are compared to any other combination listed in this survey. Finally, the X-DDPG algorithm is a version of ERL with several asynchronous DDPG actors where the buffers from the evolutionary agents and from the DDPG agents are separated, and the most recent DDPG agent is injected into the evolutionary population at each time step [20].

The Behavior-based NeuroEvolutionary Training (BNET) algorithm [92] is borderline in this survey as it does not truly use an RL algorithm, but uses a Behavior-Based Neuroevolution (BBNE) mechanism which is only loosely inspired from RL, without relying on gradient descent. BNET combines a robust selection method based on

standard fitness, a second mechanism based on the advantage of the behavior of an agent, and a third mechanism based on a surrogate estimate of the return of policies. The BBNE mechanism is reminiscent of the Advantage Weighted Regression (AWR) algorithm [70], but it uses an evolutionary approach to optimize this behavior-based criterion instead of standard gradient-based methods. The reasons for this choice is that the evolutionary part relies on Cartesian Genetic Programming (CGP) [62] which evolves the structure of the neural networks, but gradient descent operators cannot be applied to networks whose structure is evolving over episodes.

The Cooperative Heterogeneous Deep Reinforcement Learning (CHDRL) architecture [108] extends the ERL approach in several ways to improve the sample efficiency of the combination. First, it uses two levels of RL algorithms, one on-policy and one off-policy, to benefit from the higher sample efficiency of off-policy learning. Second, instead of injecting an actor periodically in the evolutionary population, it does so only when the actor to be injected performs substantially better than the evolutionary agents. Third, it combines the standard replay buffer with a smaller local one which is filled with filtered data to ensure using the most beneficial samples. The CSPC algorithm, depicted in Figure 3a is an instance of CHDRL using the CEM, SAC and PPO [81] algorithms.

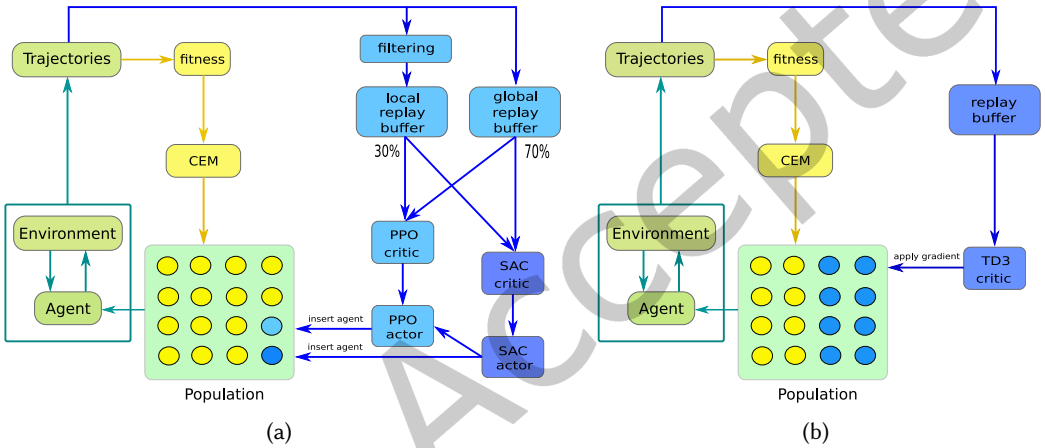

Fig. 3. The CSPC (a) and CEM-RL (b) architectures. In CSPC, an on-policy and an off-policy algorithms are combined, together with two replay buffers and a performance-based actor injection rule, to improve the sample efficiency of ERL-like methods. In CEM-RL, gradient steps from the TD3 critic are applied to half the population of evolutionary agents. If applying this gradient is favorable, the corresponding individuals are kept, otherwise they are discarded.

Note that if an RL actor is injected in an evolutionary population and if evolution uses a direct encoding, **the RL actor and evolution individuals need to share a common structure**. Removing this constraint might be useful, as evolutionary methods are often applied to smaller policies than RL methods. For doing so, one might call upon any policy distillation mechanism that strives to obtain from a large policy a smaller policy with similar capabilities.

2.2 RL gradient addition

Instead of injecting an RL actor into the population, another approach **applies gradient steps to some members of this population**. This is the approach of the CEM-RL algorithm [75], see Figure 3b, which combines the Cross-Entropy Method (CEM) [78] and Twin Delayed Deep Deterministic Policy Gradient (TD3) [22]. This work was followed by CEM-ACER [96] which simply replaces TD3 with Actor Critic with Experience Replay (ACER) [103].

2.3 Evolution from the RL actor

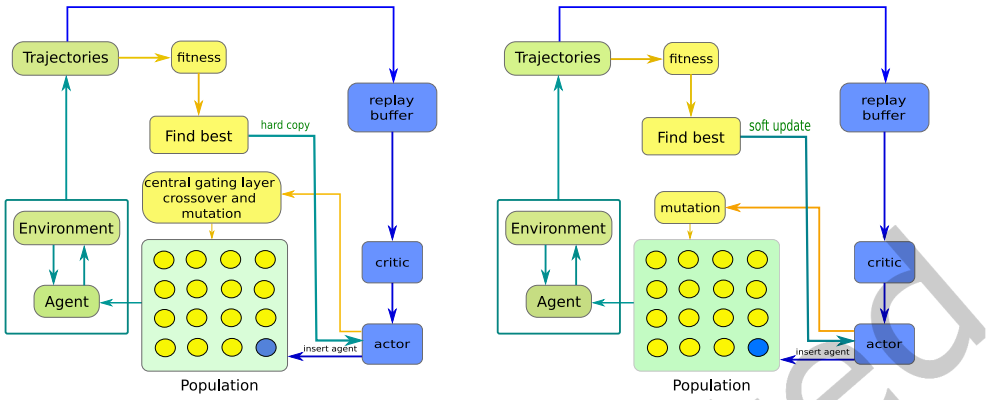

Fig. 4. In the G2N (a) and SUPE-RL (b) architectures, the evolutionary population is built locally from the RL actor. In G2N, the evolutionary part explores the structure of the central layer of the actor network. In SUPE-RL, more standard mutations are applied, the non-mutated actor is inserted in the evolutionary population and the actor is soft-updated towards its best offspring.

In the algorithms listed so far, the main loop is evolutionary and the RL loop is used at a slower pace to accelerate it. In the Genetic-Gated Networks (G2N) [5] and Soft Updates for Policy Evolution (SUPE-RL) [61] algorithms, by contrast, **the main loop is the RL loop and evolution is used to favor exploration**.

In G2N, shown in Figure 4a, evolution is used to activate or deactivate neurons of the central layer in the architecture of the actor according to a binary genome. By sampling genomes using evolutionary operators, various actor architectures are evaluated and the one that performs best benefits from a critic gradient step, before its genome is used to generate a new population of architectures. This mechanism provides a fair amount of exploration both in the actor structures and in the generated trajectories and outperforms random sampling of the genomes. Two instances of the G2N approach are studied, G2AC based on Advantage Actor Critic (A2C) [63] and G2PPO based on Proximal Policy Optimization (PPO) [81], and they both outperform the RL algorithm they use.

The SUPE-RL algorithm, shown in Figure 4b, is similar to G2N apart from the fact that evolving the structure of the central layer is replaced by performing standard Gaussian noise mutation of all the parameters of the actor. Besides, if one of the offspring is better than the current RL agent, the latter is modified towards this better offspring through a soft update mechanism. Finally, the non-mutated actor is also inserted in the evolutionary population, which is not the case in G2N.

2.4 Using a surrogate fitness

A weakness of all the methods combining evolution and RL that we have listed so far is that they require evaluating the agents to perform the evolutionary selection step, which may impair sample efficiency. In the Surrogate-Assisted Controller for ERL (SC-ERL) [102] and Coupling Policy Gradient with Population-based Search (PGPS) [41] architectures, this concern is addressed by **using a critic network as a surrogate for evaluating an agent**. Importantly, the evaluation of individuals must initially rely on the true fitness but can call upon the critic more and more often as its accuracy gets better. As shown in Figure 5a, the SC-ERL architecture is generic

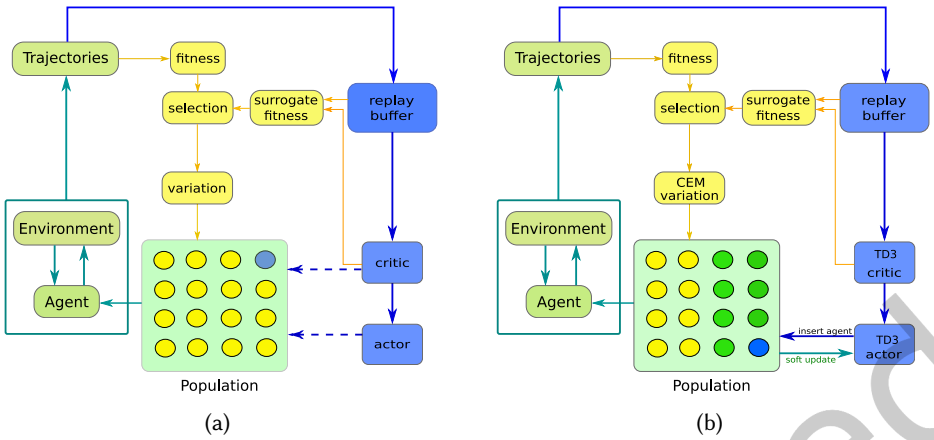

Fig. 5. The SC-ERL (a) and PGPS (b) architectures are two approaches to improve sample efficiency by using a critic network as a surrogate for evaluating evolutionary individuals. In SC-ERL, the surrogate control part is generic and can be applied to several architectures such as ERL, CERL or CEM-RL. It considers the critic as a surrogate model of fitness, making it possible to estimate the fitness of a new individual without generating additional samples. (b) The PGPS uses the same idea but combines it with several other mechanisms, such as performing a soft update of the actor towards the best evolutionary agent or filling half the population using the surrogate fitness and the other half from CEM generated agents.

and can be applied on top of any of the combinations we have listed so far. In practice, it is applied to ERL, PDERL and CEM-RL, resulting in the SERL and SPDERL algorithms in the first two cases.

The PGPS algorithm [41], shown in Figure 5b, builds on the same idea but uses it in the context of a specific combination of evolutionary and RL mechanisms which borrows ideas from several of the previously described methods. In more detail, half of the population is filled with agents evaluated from the surrogate fitness whereas the other half are generated with CEM. Furthermore, the current TD3 actor is injected into the population and benefits from a soft update towards the best agent in the population.

3 EVOLUTION OF ACTIONS FOR PERFORMANCE

In this section we cover algorithms where evolution is used **to optimize an action in a given state, rather than optimizing policy parameters**. The general idea is that variation-selection methods such as CEM can optimize any vector of parameters given some performance function of these parameters. In the methods listed in the previous section, the parameters were those of a policy and the performance was the return of that policy. In the methods listed here, the parameters specify the action in a given state and the performance is the Q-value of this action in that state.

In an RL algorithm like Q-LEARNING [104], the agent needs to find the action with the highest value in a given state for two things: for performing critic updates, that is updating its estimates of the action-value function using $Q(s_t, a_t) \leftarrow r(s_t, a_t) + \max_a Q(s_{t+1}, a) - Q(s_t, a_t)$, and for acting using $\text{argmax}_a Q(s_t, a)$. When the action space is continuous, this amounts to solving an expensive optimization problem, and this is required at each training step. The standard solution to this problem in actor-critic methods considers the action of the actor as a good proxy for the best action. The estimated best action, that we note $\bar{a}_t$, is taken to be the actor's action $\bar{a}_t = \pi(s_t)$, resulting in using $Q(s_t, a_t) \leftarrow r(s_t, a_t) + \max_a Q(s_{t+1}, \pi(s_{t+1})) - Q(s_t, a_t)$ for the critic update and using $\bar{a}_t$ for acting.

But as an alternative, one can call upon a variation-selection method to find the best performing action over a limited set of sampled actions. This approach is used in the QT-OPT algorithm [36], as well as in the Cross-Entropy Guided Policies (CGP) [90], Soft Actor-Critic with Cross-Entropy Policy Optimization (SAC-CEPO) [87], Self-Guided and Self-Regularized Actor-Critic (GRAC) [84] and Evolutionary Action Selection RL (EAS-RL) [57] algorithms. This is the approach we first cover in this section. The Zeroth-Order Supervised Policy Improvement (ZOSPI) algorithm [93] also benefits from optimizing actions with a variation-selection method, though it stems from a different perspective.

Table 2. Combinations evolving actions for performance. The cells in green denote where evolutionary optimization takes place. We specify the use of CEM for optimizing an action with $\bar{a}_t = \text{CEM}(source, N, N_e, I)$, where *source* is the source from which we sample initial actions, N is the size of this sample (the population), N_e is the number of elite solutions that are retained from a generation to the next and I is the number of iterations. For PSO, the shown parameters are the number of action N and the number of iterations T. And we use $\bar{a}_t = \text{argmax}(source, N)$ for simply take the best action over N samples from a given *source*.

Prop. / Algo.	Critic update	Action Selection	Policy Update
QT-OPT [36]	$\bar{a}_t$ =CEM (random, 64, 6, 2)	$\bar{a}_t$ =CEM (random, 64, 6, 2)	No policy
CGP [90]	$\bar{a}_t$ =CEM (random, 64, 6, 2)	$\bar{a}_t = \pi(s_t)$	BC or DPG
EAS-RL [57]	$\bar{a}_t$ =PSO (10,10)	$\bar{a}_t = \pi(s_t)$	BC + DPG
SAC-CEPO [87]	SAC update	$\bar{a}_t$ =CEM (π, 60 →140, 3% →7%, 6 →14)	BC
GRAC [84]	$\bar{a}_t$ =CEM (π, 256, 5, 2)	$\bar{a}_t$ =CEM (π, 256, 5, 2)	PG with two losses
ZOSPI [93]	DDPG update	$\bar{a}_t = \pi(s_t)$ + perturb. network	BC($\bar{a}_t$ = argmax($random$, 50))

As Table 2 shows, the QT-OPT algorithm [36] **simply samples 64 random actions** in the action space and performs two iterations of CEM to get a high performing action, both for critic updates and action selection. It is striking that such a simple method can perform well even in large action spaces. This simple idea was then improved in the CGP algorithm [90] so as to **avoid the computational cost of action inference**. Instead of using CEM to sample an action at each time step, a policy network is learned based on the behavior of the CEM. This network can be seen as a surrogate of the CEM sampling process and is trained either from the sampled $\bar{a}_t$ using Behavioral Cloning (BC) or following a Deterministic Policy Gradient (DPG) step from the critic.

The EAS-RL algorithm [57] is similar to CGP apart from the fact that it uses Particle Swarm Optimization (PSO) instead of CEM. Besides, depending on the sign of the advantage of the obtained action $\bar{a}_t$, it uses either BC or DPG to update the policy for each sample.

Symmetrically to CGP, the SAC-CEPO algorithm [87] performs standard critic updates using SAC but selects actions using CEM. More precisely, it **introduces the idea to sample the action from the current policy rather than randomly**, and updates this policy using BC from the sampled actions. Besides, the paper investigates the effect of the CEM parameters but does not provide solid conclusions.

The GRAC algorithm [84] **combines ideas from CGP and SAC-CEPO**. A stochastic policy network outputs an initial Gaussian distribution for the action at each step. Then, a step of CEM drawing 256 actions out of this distribution is used to further optimize the choice of action both for critic updates and action selection. The policy itself is updated with a combination of two training losses.

Finally, the ZOSPI algorithm [93] **calls upon variation-selection for updating the policy** rather than for updating the critic or selecting the action. Its point is rather that gradient descent algorithms tend to get stuck into local minima and may miss the appropriate direction due to various approximations, whereas a variation-selection method is more robust. Thus, to update its main policy, ZOSPI simply samples a set of actions and performs BC towards the best of these actions, which can be seen as a trivial variation-selection method. The typical number

of sampled actions is 50. It then adds a policy perturbation network to perform exploration, which is trained using gradient descent.

4 EVOLUTION OF POLICIES FOR DIVERSITY

The trade-off between exploration and exploitation is central to RL. In particular, when the reward signal is sparse, efficient exploration becomes crucial. All the papers studied in this survey manage a population of agents, hence **their capability to explore can benefit from maintaining behavioral diversity between the agents**. This idea of maintaining behavioral diversity is central to two families of diversity seeking algorithms, the novelty search (NS) [46] algorithms which do not use the reward signal at all, see Figure 6a, and the quality-diversity (QD) algorithms [15, 76], see Figure 6b, which try to maximize both diversity and performance. As the NS approach only looks for diversity, it is better in the absence of reward, or when the reward signal is very sparse or deceptive as the best one can do in the absence of reward is try to cover a relevant space as uniformly as possible [18]. By contrast, the QD approach is more appropriate when the reward signal can contribute to the policy search process. In this section we cover both families separately.

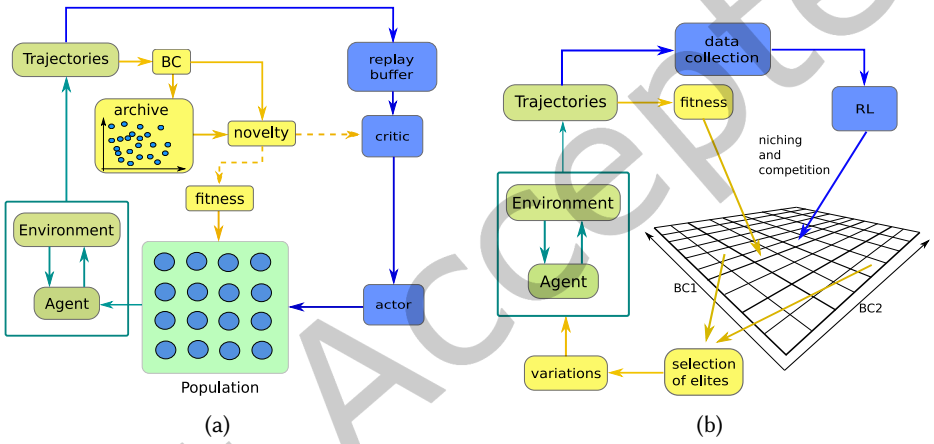

Fig. 6. Template architectures for combining deep RL with novelty search (a) and quality-diversity (b). The latter builds on Fig. 2 in [65]. Both architectures rely on a behavioral characterization space and maintain an archive in that space.

4.1 Novelty seeking approaches

Maintaining a distance between agents in a population can be achieved in different spaces. For instance, the Stein Variational Policy Gradient (SVPG) algorithm [54] defines distances in a kernel space and adds to the policy gradient a loss term dedicated to increasing the pairwise distance between agents. Alternatively, the Diversity via Determinants (DVD) algorithm [68] defines distances in an action embedding space, corresponding to the actions specified by each agent in a large enough set of random states. Then DVD optimizes a global distance between all policies by maximizing the volume of the space between them through the computation of a determinant. Despite their interest, these two methods depicted in Figure 7 do not appear in Table 3 as the former does not have an evolutionary component and the latter uses NSR-ES [14] but does not have an RL component.

A more borderline case with respect to the focus of this survey is the Population-guided Parallel Policy Search (P3S-TD3) algorithm [35]. Though P3S-TD3 is used as a baseline in several of the papers mentioned in this survey, its equivalent of the evolutionary loop is limited to finding the best agent in the population, as shown in Figure 8a.

Table 3. Combinations evolving policies for diversity. NS: Novelty Search. Policy params: distance is computed in the policy parameters space. GC-DDPG: goal-conditioned DDPG. Manual BC: distances are computed in a manually defined behavior characterization space.

Prop. / Algo.	RL algo.	Diversity algo.	Distance space
P3S-TD3 [35]	TD3	Find best	Policy params.
DEPRL [52]	TD3	CEM	Policy params.
ARAC [17]	SAC	NS-like	Policy params.
NS-RL [86]	GC-DDPG	True NS	Manual BC
PNS-RL [53]	TD3	True NS	Manual BC

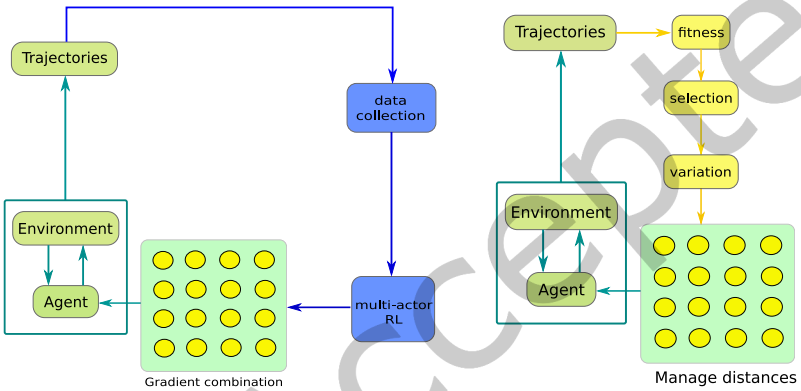

Fig. 7. The SVPG (a) and DVD (b) architectures. In SVPG, individual policy gradients computed by each agent are combined so as to ensure both diversity between agents and performance improvement. In DVD, a purely evolutionary approach is combined with a diversity mechanism which seeks to maximize the volume between the behavioral characterization of agents in an action embedding space. Both architectures do not aim to combine evolution and RL, though they both try to maximize diversity and performance in a population of agents.

This implies evaluating all these agents, but not using neither variation nor selection. Besides, the mechanism to maintain a distance between solutions in P3S-TD3 is ad hoc and acts in the space of policy parameters. This is also the case in the Diversity Evolutionary Policy Deep Reinforcement Learning (DEPRL) algorithm [52], which is just a variation of CEM-RL where an ad hoc mechanism is added to enforce some distance between members of the evolutionary population.

The Attraction-Repulsion Actor-Critic (ARAC) algorithm [17] also uses a distance in the policy parameter space, but it truly qualifies as a combination of evolution and deep RL, see Figure 8b. An original feature of ARAC is that it selects the data coming into the replay buffer based on the novelty of agents, which can result in saving a lot of poorly informative gradient computations. A similar idea is also present in [6] where instead of filtering based on novelty, the algorithm uses an elite replay buffer containing only the top trajectories, similarly to what we have already seen in the CSPC algorithm [108].

The methods listed so far were neither using a behavior characterization space for computing distances between agents nor an archive of previous agents to evaluate novelty. Thus they do not truly qualify as NS approaches. We now turn to algorithms which combine both features.

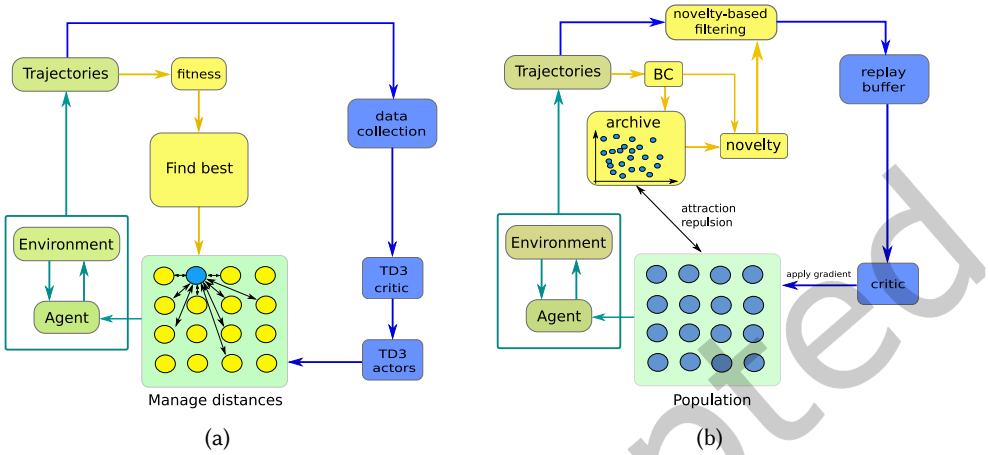

Fig. 8. The P3S-TD3 (a) and ARAC (b) architectures. In P3S-TD3, all agents are trained with RL and evaluated, then they all perform a soft update towards the best agent. The ARAC algorithm maintains a population of policies following a gradient from a common SAC critic [27]. The critic itself is trained from trajectories of the most novel agents. Besides, diversity in the population is ensured by adding an attraction-repulsion loss $\mathcal{L}_{AR}$ to the update of the agents. This loss is computed with respect to an archive of previous agents themselves selected using a novelty criterion.

Figure 6a suggests that, when combining evolution and RL, novelty can be used as a fitness function, as a reward signal to learn a critic, or both. Actually, in the two algorithms described below, it is used for both. More precisely, the RL part is used to move the rest of the population towards the most novel agent.

In the Population-Guided Novelty Search for Reinforcement Learning (PNS-RL) algorithm [53], a group of agents is following a leader combining a standard policy gradient update and a soft update towards the leader. Then, for any agent in the group, if its performance is high enough with respect to the mean performance in an archive, it is added to the archive. Crucially, **the leader is selected as the one that maximizes novelty** in the archive given a manually defined behavioral characterization. In addition, for efficient parallelization, the algorithm considers several groups instead of one, but where all groups share the same leader.

The Novelty Search RL (NS-RL) algorithm [86] can be seen as a version of CEM-RL whose RL **part targets higher novelty** by training less novel agents to minimize in each step the distance to the BC of the most novel agent. As the most novel agent and its BC change in each iteration, **the RL part is implemented with goal-conditioned policies**. This implies that the goal space is identical to the behavioral characterization space.

4.2 Quality-diversity approaches

By contrast with NS approaches which only try to optimize diversity in the population, QD approaches combine this first objective with optimize the performance of registered policies, their *quality*. As Figure 6b suggests, when combined with an RL loop, the QD loop can give rise to various solutions depending on whether quality and diversity are improved with an evolutionary algorithm or a deep RL algorithm.

The space of resulting possibilities is covered in Table 4. In more details, the Policy Gradient Assisted MAP-Elites (PGA-ME) algorithm [66] uses two optimization mechanisms, TD3 and a GA, to generate new solutions that

Table 4. Quality-Diversity algorithms including an RL component. All these algorithms rely on the MAP-Elites approach and the BC space is defined manually. For each algorithm in the rows, the table states whether quality and diversity are optimized using an RL approach or an evolutionary approach.

Prop. / Algo.	Type of Archive	Q. improvement	D. improvement
PGA-ME [66]	MAP-Elites	TD3 or GA	TD3 or GA
QD-PG-PF [11]	Pareto front	TD3	TD3
QD-PG-ME [71]	MAP-Elites	TD3	TD3
CMA-MEGA-(TD3, ES) [99]	MAP-Elites	TD3 + OPENAI-ES	OPENAI-ES

are added to the archive if they are either novel enough or more efficient than previously registered ones with the same behavioral characterization. By contrast, in the Quality-Diversity RL (QD-RL) approach, the mechanisms to improve quality and diversity are explicitly separated and improve a quality critic and a diversity critic using TD3. Two implementations exist. First, the QD-PG-PF algorithm [11] maintains a Pareto front of high quality and diversity solutions. From its side, the QD-PG-ME algorithm [71] maintains a MAP-Elites archive and introduces an additional notion of state descriptor to justify learning a state-based quality critic. Finally, the Covariance Matrix Adaptation MAP-Elites via a Gradient Arborescence (CMA-MEGA) approach [99] uses the OPENAI-ES algorithm [79] to improve diversity and either OPENAI-ES or a combination of OPENAI-ES and RL to improve quality. Table 4 only shows the latter. Note that, by contrast to the other algorithms, the combination mechanism comes with additional parameters, which are themselves optimized with CMA-ES.

To summarize, one can see that both quality and diversity can be improved through RL, evolution, or both.

5 EVOLUTION OF SOMETHING ELSE

In all the architecture we have surveyed so far, the evolutionary part was used to optimize either policy parameters or a set of rewarding actions in a given state. In this section, we briefly cover combinations of evolution and deep RL where evolution is used to optimize something else that matters in the RL process, or where RL mechanisms are used to improve evolution without calling upon a full RL algorithm. We dedicate a separate part to optimizing hyperparameters, as it is an important and active domain.

5.1 Evolution in MBRL

The CEM algorithm can be used to optimize open-loop controllers to perform Model Predictive Control (MPC) on robotic systems in the Deep Planning Network (PLANET) [28] and Policy Planning (POPLIN) [101] algorithms, and an improved version of CEM for this specific context is proposed in [73, 74]. Besides, this approach combining open-loop controllers and MPC is seen in the Probabilistic Ensembles with Trajectory Sampling (PETS) algorithm [10] as implementing a form of Model-Based Reinforcement Learning (MBRL), and CEM is used in PETS to choose the points from where to start MPC, improving over random shooting. Finally, in [3], the authors propose to interleave CEM iterations and Stochastic Gradient Descent (SGD) iterations to improve the efficiency of optimization of **MPC plans**, in a way reminiscent to CEM-RL combining policy gradient steps and CEM steps. But all these methods are applied to an open-loop control context where true reinforcement learning algorithms can not be applied, hence they do not appear in Table 5.

Table 5. Algorithms where evolution is applied to something else than action or policy parameters, or to more than policy parameters. All algorithms in the first half optimize hyperparameters. *: The algorithm in [67] is given no name. BT stands for Behavior Tree.

Prop. / Algo.	RL algo.	Evo algo.	Evolves what?
GA-DRL [82, 83]	DDPG (+HER)	GA	Hyper-parameters
PBT [34]	Any	Ad hoc	Parameters and Hyper-parameters
AAC [24]	SAC	Ad hoc	Parameters and Hyper-parameters
SEARL [21]	TD3	GA	Architecture, Parameters and Hyper-parameters
OHT-ES [97]	Any	ES	Hyper-parameters
EPG [32]	Ad hoc (~ PPO)	ES	Reward-related functions
EQ [48]	~ DDPG	~ CEM	Critic
EVO-RL [30]	Q-LEARNING, DQN, PPO	BT	Partial policies
DERL [25]	PPO	GA	System's morphology
* [67]	PPO	GA	System's morphology

5.2 Evolution of hyper-parameters

Hyperparameter optimization (HPO) is notoriously hard and often critical in deep RL. The most straightforward way to leverage evolutionary methods in this context nests the deep RL algorithm within an evolutionary loop which tunes the hyper-parameters. This is the approach of the Genetic Algorithm Deep RL (GA-DRL) algorithm [82, 83], but this obviously suffers from a very high computational cost. Note that the authors write that GA-DRL uses DDPG + Hindsight Experience replay (HER) [1], but the use of HER is in no way clear as the algorithm does not seem to use goal-conditioned policies.

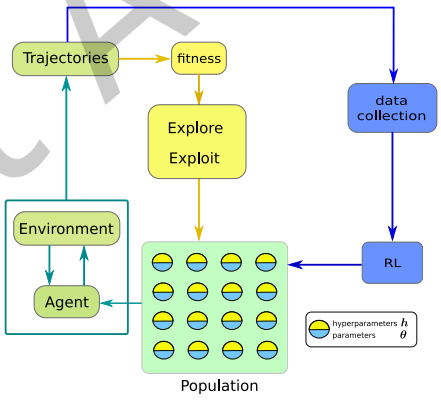

Fig. 9. The PBT architecture. The evolution part consists of two operators, *explore* and *exploit* which act both on the hyperparameters and the parameters of the agents.

More interestingly, the Population-Based Training (PBT) architecture [34] is designed to solve this problem by combining distributed RL with an evolutionary mechanism which acts both on the parameters and hyperparameters within the RL training loop. It was successfully used in several challenging applications [33] and benefits from

an interesting capability to **tune the hyperparameters according to the current training dynamics**, which is an important meta-learning capability [39]. A follow-up of the PBT algorithm is the Automatic Actor-Critic (AAC) algorithm [24], which basically applies the same approach but with a better set of hyperparameters building on lessons learned in the recent deep RL literature.

A limitation of PBT is that each actor uses its own replay buffer. Instead, in the Sample-Efficient Automated Deep Reinforcement Learning (SEARL) algorithm [21], **the experience of all agents is shared into a unique buffer**. Furthermore, SEARL**simultaneously performs HPO and Neural Architecture Search**, resulting in better performance than PBT. Finally, the Online Hyper-parameter Tuning via Evolutionary Strategies (OHT-ES) algorithm [97] also uses a shared replay buffer, but limits the role of evolution to optimizing hyperparameters and does so with an ES algorithm. Given the importance of the problem, there are many other HPO methods, most of which are not explicitly calling upon an evolutionary approach. For a wider survey of the topic, we refer the reader to [69].

5.3 Evolution of miscellaneous RL or control components

Finally, we briefly survey the rest of algorithms listed in Table 5. The Evolved Policy Gradient (EPG) algorithm [32] uses a meta-learning approach to evolve **the parameters of a loss function** that replaces the policy gradient surrogate loss in policy gradient algorithms. The goal is to find a reward function that will maximize the capability of an RL algorithm to achieve a given task. A consequence of its design is that it cannot be applied to an actor-critic approach.

Instead of evolving a population of agents, the Evolved Q-maps (EQ) algorithm [48] evolves **a population of critics**, which are fixed over the course of learning for a given agent. This is somewhat symmetric to the Zeroth-Order Actor-Critic (ZOAC) algorithm [47] which uses evolution to update an actor given a critic trained with RL.

The Evolutionary-driven RL (EVO-RL) algorithm [29] evolves **partial policies**. Evolution is performed in a discrete action context with a Genetic Programming approach [42] that only specifies a partial policy as Behavior Trees (BTs) [13]. An RL algorithm such as Deep Q-Network (DQN) [64] or PPO is then in charge of learning a policy for the states for which an action is not specified. The fitness of individuals is evaluated over their overall behavior combining the BT part and the learned part, but only the BT part is evolved to generate the next generation, benefiting from a Baldwin effect [91].

Finally, several works consider evolving **the morphology of a mechanical system** whose control is learned with RL. Table 5 only mentions two recent instances, one where the algorithm is not named [25] and Deep Evolutionary Reinforcement Learning (DERL) [67], but this idea has led to a larger body of works, e.g. [26, 56].

5.4 Evolution improved with RL mechanisms

Without using a full RL part, a few algorithms augment an evolutionary approach with components taken from RL.

First, the Trust Region Evolution Strategies (TRES) algorithm [51] incorporates into an ES several ideas from the TRPO [80] and PPO [81] algorithms, such as introducing an importance sampling mechanism and using a clipped surrogate objective so as to enforce a natural gradient update. Unfortunately, TRES is neither compared to the NES algorithm [106] which also enforces a natural gradient update nor to the safe mutation mechanism of [45] which has similar properties.

Second, there are two perspectives about the previously mentioned ZOAC algorithm [47]. One can see it as close to the ZOSPI algorithm described in Section 3, that is an actor-critic method where gradient descent to update the actor given the critic is replaced by a more robust derivative-free approach. But the more accurate perspective, as put forward by the authors, is that ZOAC is an ES method where the standard ES gradient estimator is replaced by

a gradient estimator using the advantage function so as to benefit from the capabilities of the temporal difference methods to efficiently deal with the temporal credit assignment problem.

Finally, with their GUIDED ES algorithm [58], the authors study how a simple ES gradient estimator can be improved by leveraging knowledge of an approximate gradient suffering from bias and variance. Though their study is general, it is natural to apply it to the context where the approximate gradient is a policy gradient, in which case GUIDED ES combines evolution and RL. This work is often cited in a very active recent trend which tries to improve the exploration capabilities of ES algorithms by drawing better than Gaussian directions to get a more informative gradient approximator [7, 8, 16, 107]. In particular, the SGES algorithm [50] leverages both the GUIDED ES idea and the improved exploration ideas to produce a competitive ES-based policy search algorithm.

6 CONCLUSION

In this paper we have provided a list of all the algorithms combining evolutionary processes and deep reinforcement learning we could find, irrespective of the publication status of the corresponding papers. Our focus was on the mechanisms and our main contribution was to provide a categorization of these algorithms into several groups of methods, based on the role of evolutionary optimization in the architecture.

We have not covered related fields such as algorithms which combine deep RL and imitation learning, though at least one of them also includes evolution [55]. Besides, we have not covered works which focus on the implementation of evolution and deep RL combinations, such as [44] which shows the importance of asynchronism in such combinations.

Despite these limitations, the scope of the survey was still too broad to enable deeper analyses of the different combination methods or a comparative evaluation of their performance. In the future, we intend to focus separately on the different categories so as to provide these more in-depth analyses and perform comparative evaluation of these algorithms between each other and with respect to state of the art deep RL algorithms, based on a unified benchmark.

Our focus on elementary mechanisms also suggests the possibility to design new combinations of such mechanisms, that is **combining the combinations**. For instance, one may include into a single architecture the idea of selecting samples sent to the replay buffer so as to maximize the efficiency of the RL component, more efficient crossover or mutation operators as in PDERL, soft policy updates, hyperparameter tuning etc. No doubt that such combinations will emerge in the future if they can result in additional performance gains, despite the additional implementation complexity.

ACKNOWLEDGMENTS

The author wants to thank three anonymous reviewers. In addition, the author would like to thank Giuseppe Paolo, Stéphane Doncieux and Antonin Raffin for useful remarks about this manuscript as well as several colleagues from ISIR for their questions and remarks about the algorithms.

REFERENCES

[1] Marcin Andrychowicz, Dwight Crow, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 5048–5058. https://proceedings.neurips.cc/paper/2017/hash/453fadbd8a1a3af50a9df4df899537b5-Abstract.html

[2] Thomas Bäck, Ulrich Hammel, and H.-P. Schwefel. 1997. Evolutionary computation: Comments on the history and current state. *IEEE transactions on Evolutionary Computation* 1, 1 (1997), 3–17.

[3] Homanga Bharadhwaj, Kevin Xie, and Florian Shkurti. 2020. Model-predictive control via cross-entropy and gradient-based optimization. In *Learning for Dynamics and Control*. PMLR, 277–286.

[4] Cristian Bodnar, Ben Day, and Pietro Lió. 2020. Proximal Distilled Evolutionary Reinforcement Learning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 3283–3290. https://aaai.org/ojs/index.php/AAAI/article/view/5728

[5] Simyung Chang, John Yang, Jaeseok Choi, and Nojun Kwak. 2018. Genetic-Gated Networks for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 1754–1763. https://proceedings.neurips.cc/paper/2018/hash/d516b13671a4179d9b7b458a6ebdeb92-Abstract.html

[6] Gang Chen. 2019. Merging Deterministic Policy Gradient Estimations with Varied Bias-Variance Tradeoff for Effective Deep Reinforcement Learning. *ArXiv preprint* abs/1911.10527 (2019). https://arxiv.org/abs/1911.10527

[7] Krzysztof Choromanski, Aldo Pacchiano, Jack Parker-Holder, Yunhao Tang, and Vikas Sindhwani. 2019. From Complexity to Simplicity: Adaptive ES-Active Subspaces for Blackbox Optimization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 10299–10309. https://proceedings.neurips.cc/paper/2019/hash/88bade49e98db8790df275fcebb37a13-Abstract.html

[8] Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard E. Turner, and Adrian Weller. 2018. Structured Evolution with Compact Architectures for Scalable Policy Optimization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer G. Dy and Andreas Krause (Eds.). PMLR, 969–977. http://proceedings.mlr.press/v80/choromanski18a.html

[9] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. 2018. Back to Basics: Benchmarking Canonical Evolution Strategies for Playing Atari. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, Jérôme Lang (Ed.). ijcai.org, 1419–1426. https://doi.org/10.24963/ijcai.2018/197

[10] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. 2018. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems* 31 (2018).

[11] Geoffrey Cideron, Thomas Pierrot, Nicolas Perrin, Karim Beguir, and Olivier Sigaud. 2020. QD-RL: Efficient Mixing of Quality and Diversity in Reinforcement Learning. *ArXiv preprint* abs/2006.08505 (2020). https://arxiv.org/abs/2006.08505

[12] Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. 2018. GEP-PG: Decoupling Exploration and Exploitation in Deep Reinforcement Learning Algorithms. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer G. Dy and Andreas Krause (Eds.). PMLR, 1038–1047. http://proceedings.mlr.press/v80/colas18a.html

[13] Michele Colledanchise and Petter Ögren. 2018. *Behavior trees in robotics and AI: An introduction*. CRC Press.

[14] Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. 2018. Improving Exploration in Evolution Strategies for Deep Reinforcement Learning via a Population of Novelty-Seeking Agents. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 5032–5043. https://proceedings.neurips.cc/paper/2018/hash/b1301141feffabac455e1f90a7de2054-Abstract.html

[15] Antoine Cully and Yiannis Demiris. 2017. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation* 22, 2 (2017), 245–259.

[16] Anton Dereventsov, Clayton G. Webster, and Joseph Daws. 2022. An adaptive stochastic gradient-free approach for high-dimensional blackbox optimization. In *Proceedings of International Conference on Computational Intelligence*. Springer, 333–348.

[17] Thang Doan, Bogdan Mazoure, Audrey Durand, Joelle Pineau, and R Devon Hjelm. 2019. Attraction-Repulsion Actor-Critic for Continuous Control Reinforcement Learning. *ArXiv preprint* abs/1909.07543 (2019). https://arxiv.org/abs/1909.07543

[18] Stephane Doncieux, Alban Laflaquière, and Alexandre Coninx. 2019. Novelty search: a theoretical perspective. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 99–106.

[19] Madalina M. Drugan. 2019. Reinforcement learning versus evolutionary computation: A survey on hybrid algorithms. *Swarm and evolutionary computation* 44 (2019), 228–246.

[20] Federico Espositi and Andrea Bonarini. 2020. Gradient Bias to Solve the Generalization Limit of Genetic Algorithms Through Hybridization with Reinforcement Learning. In *International Conference on Machine Learning, Optimization, and Data Science*. Springer, 273–284.

[21] Jörg K. H. Franke, Gregor Köhler, André Biedenkapp, and Frank Hutter. 2021. Sample-Efficient Automated Deep Reinforcement Learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=hSjxQ3B7GWq

[22] Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer G. Dy and Andreas Krause (Eds.). PMLR, 1582–1591. http:

//proceedings.mlr.press/v80/fujimoto18a.html
[23] Tanmay Gangwani and Jian Peng. 2018. Policy Optimization by Genetic Distillation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=ByOnmlWC-
[24] Jake Grigsby, Jin Yong Yoo, and Yanjun Qi. 2021. Towards Automatic Actor-Critic Solutions to Continuous Control. *ArXiv preprint* abs/2106.08918 (2021). https://arxiv.org/abs/2106.08918
[25] Agrim Gupta, Silvio Savarese, Surya Ganguli, and Li Fei-Fei. 2021. Embodied Intelligence via Learning and Evolution. *ArXiv preprint* abs/2102.02202 (2021). https://arxiv.org/abs/2102.02202
[26] David Ha. 2019. Reinforcement learning for improving agent design. *Artificial life* 25, 4 (2019), 352–365.
[27] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. 2018. Soft actor-critic algorithms and applications. *ArXiv preprint* abs/1812.05905 (2018). https://arxiv.org/abs/1812.05905
[28] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2019. Learning latent dynamics for planning from pixels. In *International conference on machine learning*. PMLR, 2555–2565.
[29] Ahmed Hallawa, Thorsten Born, Anke Schmeink, Guido Dartmann, Arne Peine, Lukas Martin, Giovanni Iacca, AE Eiben, and Gerd Ascheid. 2021. Evo-RL: evolutionary-driven reinforcement learning. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 153–154.
[30] Ahmed Hallawa, Jaro De Roose, Martin Andraud, Marian Verhelst, and Gerd Ascheid. 2017. Instinct-driven dynamic hardware reconfiguration: evolutionary algorithm optimized compression for autonomous sensory agents. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 1727–1734.
[31] John H Holland and Judith S Reitman. 1978. Cognitive systems based on adaptive algorithms. In *Pattern-directed inference systems*. Elsevier, 313–329.
[32] Rein Houthooft, Yuhua Chen, Phillip Isola, Bradly C. Stadie, Filip Wolski, Jonathan Ho, and Pieter Abbeel. 2018. Evolved Policy Gradients. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 5405–5414. https://proceedings.neurips.cc/paper/2018/hash/7876acb66640bad41f1e1371ef30c180-Abstract.html
[33] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. 2019. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science* 364, 6443 (2019), 859–865.
[34] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. 2017. Population-based training of neural networks. *ArXiv preprint* abs/1711.09846 (2017). https://arxiv.org/abs/1711.09846
[35] Whiyoung Jung, Giseung Park, and Youngchul Sung. 2020. Population-Guided Parallel Policy Search for Reinforcement Learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=rJeINp4KwH
[36] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. 2018. QT-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *ArXiv preprint* abs/1806.10293 (2018). https://arxiv.org/abs/1806.10293
[37] Shauharda Khadka, Somdeb Majumdar, Tarek Nassar, Zach Dwiel, Evren Tumer, Santiago Miret, Yinyin Liu, and Kagan Tumer. 2019. Collaborative Evolutionary Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 3341–3350. http://proceedings.mlr.press/v97/khadka19a.html
[38] Shauharda Khadka and Kagan Tumer. 2018. Evolution-Guided Policy Gradient in Reinforcement Learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 1196–1208. https://proceedings.neurips.cc/paper/2018/hash/85fc37b18c57097425b52fc7afbb6969-Abstract.html
[39] Mehdi Khamassi, George Velentzas, Theodore Tsitsimis, and Costas Tzafestas. 2017. Active exploration and parameterized reinforcement learning applied to a simulated human-robot interaction task. In *2017 First IEEE International Conference on Robotic Computing (IRC)*. IEEE, 28–35.
[40] Kyung-Joong Kim, Heejin Choi, and Sung-Bae Cho. 2007. Hybrid of evolution and reinforcement learning for othello players. In *2007 IEEE Symposium on Computational Intelligence and Games*. IEEE, 203–209.
[41] Namyong Kim, Hyunsuk Baek, and Hayong Shin. 2020. PGPS: Coupling Policy Gradient with Population-based Search. In *Submitted to ICLR 2021*.
[42] John R. Koza et al. 1994. *Genetic programming II*. Vol. 17. MIT press Cambridge.
[43] Pier Luca Lanzi. 1999. An analysis of generalization in the XCS classifier system. *Evolutionary computation* 7, 2 (1999), 125–149.

[44] Kyunghyun Lee, Byeong-Uk Lee, Ukcheol Shin, and In So Kweon. 2020. An Efficient Asynchronous Method for Integrating Evolutionary and Gradient-based Policy Search. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/731309c4bb223491a9f67eac5214fb2e-Abstract.html

[45] Joel Lehman, Jay Chen, Jeff Clune, and Kenneth O Stanley. 2018. Safe mutations for deep and recurrent neural networks through output gradients. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 117–124.

[46] Joel Lehman and Kenneth O Stanley. 2011. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation* 19, 2 (2011), 189–223.

[47] Yuheng Lei, Jianyu Chen, Shengbo Eben Li, and Sifa Zheng. 2022. Zeroth-Order Actor-Critic. *ArXiv preprint* abs/2201.12518 (2022). https://arxiv.org/abs/2201.12518

[48] Abe Leite, Madhavun Candadai, and Eduardo J Izquierdo. 2020. Reinforcement learning beyond the Bellman equation: Exploring critic objectives using evolution. In *ALIFE 2020: The 2020 Conference on Artificial Life*.

[49] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1509.02971

[50] Fei-Yu Liu, Zi-Niu Li, and Chao Qian. 2020. Self-Guided Evolution Strategies with Historical Estimated Gradients. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, Christian Bessiere (Ed.). ijcai.org, 1474–1480. https://doi.org/10.24963/ijcai.2020/205

[51] Guoqing Liu, Li Zhao, Feidiao Yang, Jiang Bian, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2019. Trust Region Evolution Strategies. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 4352–4359. https://doi.org/10.1609/aaai.v33i01.33014352

[52] Jian Liu and Liming Feng. 2021. Diversity Evolutionary Policy Deep Reinforcement Learning. *Computational Intelligence and Neuroscience* 2021 (2021).

[53] Qihao Liu, Yujia Wang, and Xiaofeng Liu. 2018. PNS: Population-Guided Novelty Search for Reinforcement Learning in Hard Exploration Environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5627–5634.

[54] Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. 2017. Stein Variational Policy Gradient. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*, Gal Elidan, Kristian Kersting, and Alexander T. Ihler (Eds.). AUAI Press. http://auai.org/uai2017/proceedings/papers/239.pdf

[55] Shuai Lü, Shuai Han, Wenbo Zhou, and Junwei Zhang. 2021. Recruitment-imitation mechanism for evolutionary reinforcement learning. *Information Sciences* 553 (2021), 172–188.

[56] Kevin Sebastian Luck, Heni Ben Amor, and Roberto Calandra. 2020. Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning. In *Conference on Robot Learning*. PMLR, 854–869.

[57] Yan Ma, Tianxing Liu, Bingsheng Wei, Yi Liu, Kang Xu, and Wei Li. 2022. Evolutionary Action Selection for Gradient-based Policy Learning. *ArXiv preprint* abs/2201.04286 (2022). https://arxiv.org/abs/2201.04286

[58] Niru Maheswaranathan, Luke Metz, George Tucker, Dami Choi, and Jascha Sohl-Dickstein. 2019. Guided evolutionary strategies: augmenting random search with surrogate gradients. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 4264–4273. http://proceedings.mlr.press/v97/maheswaranathan19a.html

[59] Amjad Yousef Majid, Serge Saaybi, Tomas van Rietbergen, Vincent Francois-Lavet, R. Venkatesha Prasad, and Chris Verhoeven. 2021. Deep Reinforcement Learning Versus Evolution Strategies: A Comparative Survey. *ArXiv preprint* abs/2110.01411 (2021). https://arxiv.org/abs/2110.01411

[60] Horia Mania, Aurelia Guy, and Benjamin Recht. 2018. Simple random search of static linear policies is competitive for reinforcement learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 1805–1814. https://proceedings.neurips.cc/paper/2018/hash/7634ea65a4e6d9041cfd3f7de18e334a-Abstract.html

[61] Enrico Marchesini, Davide Corsi, and Alessandro Farinelli. 2021. Genetic Soft Updates for Policy Evolution in Deep Reinforcement Learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=TGFO0DbD_pk

[62] Julian Francis Miller and Simon L Harding. 2009. Cartesian genetic programming. In *Proceedings of the 11th annual conference companion on genetic and evolutionary computation conference: late breaking papers*. 3489–3512.

[63] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*. PMLR, 1928–1937.

[64] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[65] Jean-Baptiste Mouret. 2020. Evolving the behavior of machines: from micro to macroevolution. *Iscience* 23, 11 (2020), 101731.

[66] Olle Nilsson and Antoine Cully. 2021. Policy gradient assisted MAP-Elites. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 866–875.

[67] Jai Hoon Park and Kang Hoon Lee. 2021. Computational Design of Modular Robots Based on Genetic Algorithm and Reinforcement Learning. *Symmetry* 13, 3 (2021), 471.

[68] Jack Parker-Holder, Aldo Pacchiano, Krzysztof Marcin Choromanski, and Stephen J. Roberts. 2020. Effective Diversity in Population Based Reinforcement Learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/d1dc3a8270a6f9394f88847d7f0050cf-Abstract.html

[69] Jack Parker-Holder, Raghu Rajan, Xingyou Song, André Biedenkapp, Yingjie Miao, Theresa Eimer, Baohe Zhang, Vu Nguyen, Roberto Calandra, Aleksandra Faust, et al. 2022. Automated Reinforcement Learning (AutoRL): A Survey and Open Problems. *ArXiv preprint* abs/2201.03916 (2022). https://arxiv.org/abs/2201.03916

[70] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. 2019. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *ArXiv preprint* abs/1910.00177 (2019). https://arxiv.org/abs/1910.00177

[71] Thomas Pierrot, Valentin Macé, Geoffrey Cideron, Nicolas Perrin, Karim Beguir, and Olivier Sigaud. 2020. Sample efficient Quality Diversity for neural continuous control. *unpublished* (2020).

[72] Thomas Pierrot, Nicolas Perrin, and Olivier Sigaud. 2018. First-order and second-order variants of the gradient descent in a unified framework. *ArXiv preprint* abs/1810.08102 (2018). https://arxiv.org/abs/1810.08102

[73] Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. 2020. Sample-efficient cross-entropy method for real-time planning. *ArXiv preprint* abs/2008.06389 (2020). https://arxiv.org/abs/2008.06389

[74] Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, and Georg Martius. 2021. Extracting Strong Policies for Robotics Tasks from Zero-Order Trajectory Optimizers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=Nc3TJqbcl3

[75] Aloïs Pourchot and Olivier Sigaud. 2019. CEM-RL: Combining evolutionary and gradient-based methods for policy search. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=BkeU5j0ctQ

[76] Justin K. Pugh, Lisa B. Soros, and Kenneth O. Stanley. 2016. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI* 3 (2016), 40.

[77] Hong Qian and Yang Yu. 2021. Derivative-free reinforcement learning: A review. *ArXiv preprint* abs/2102.05710 (2021). https://arxiv.org/abs/2102.05710

[78] Reuven Rubinstein. 1999. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability* 1, 2 (1999), 127–190.

[79] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *ArXiv preprint* abs/1703.03864 (2017). https://arxiv.org/abs/1703.03864

[80] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015 (JMLR Workshop and Conference Proceedings, Vol. 37)*, Francis R. Bach and David M. Blei (Eds.). JMLR.org, 1889–1897. http://proceedings.mlr.press/v37/schulman15.html

[81] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *ArXiv preprint* abs/1707.06347 (2017). https://arxiv.org/abs/1707.06347

[82] Adarsh Sehgal, Hung La, Sushil Louis, and Hai Nguyen. 2019. Deep reinforcement learning using genetic algorithm for parameter optimization. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, 596–601.

[83] Adarsh Sehgal, Nicholas Ward, Hung Manh La, Christos Papachristos, and Sushil Louis. 2022. GA-DRL: Genetic Algorithm-Based Function Optimizer in Deep Reinforcement Learning for Robotic Manipulation Tasks. *ArXiv preprint* abs/2203.00141 (2022). https://arxiv.org/abs/2203.00141

[84] Lin Shao, Yifan You, Mengyuan Yan, Shenli Yuan, Qingyun Sun, and Jeannette Bohg. 2021. GRAC: Self-guided and self-regularized actor-critic. In *Conference on Robot Learning*. PMLR, 267–276.

[85] Longxiang Shi, Shijian Li, Longbing Cao, Long Yang, Gang Zheng, and Gang Pan. 2019. FiDi-RL: Incorporating Deep Reinforcement Learning with Finite-Difference Policy Search for Efficient Learning of Continuous Control. *ArXiv preprint* abs/1907.00526 (2019). https://arxiv.org/abs/1907.00526

[86] Longxiang Shi, Shijian Li, Qian Zheng, Min Yao, and Gang Pan. 2020. Efficient novelty search through deep reinforcement learning. *IEEE Access* 8 (2020), 128809–128818.

[87] Zhenyang Shi and Surya PN Singh. 2021. Soft Actor-Critic with Cross-Entropy Policy Optimization. *ArXiv preprint* abs/2112.11115 (2021). https://arxiv.org/abs/2112.11115
[88] Olivier Sigaud and Freek Stulp. 2019. Policy Search in Continuous Action Domains: an Overview. *Neural Networks* 113 (2019), 28–40.
[89] Olivier Sigaud and S. W. Wilson. 2007. Learning Classifier Systems: A Survey. *Journal of Soft Computing* 11, 11 (2007), 1065–1078.
[90] Riley Simmons-Edler, Ben Eisner, Eric Mitchell, Sebastian Seung, and Daniel Lee. 2019. Q-learning for continuous actions with cross-entropy guided policies. *ArXiv preprint* abs/1903.10605 (2019). https://arxiv.org/abs/1903.10605
[91] George Gaylord Simpson. 1953. The baldwin effect. *Evolution* 7, 2 (1953), 110–117.
[92] Jörg Stork, Martin Zaefferer, Nils Eisler, Patrick Tichelmann, Thomas Bartz-Beielstein, and AE Eiben. 2021. Behavior-based neuroevolutionary training in reinforcement learning. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 1753–1761.
[93] Hao Sun, Ziping Xu, Yuhang Song, Meng Fang, Jiechao Xiong, Bo Dai, and Bolei Zhou. 2020. Zeroth-order supervised policy improvement. *ArXiv preprint* abs/2006.06600 (2020). https://arxiv.org/abs/2006.06600
[94] Karush Suri, Xiao Qi Shi, Konstantinos N. Plataniotis, and Yuri A. Lawryshyn. 2020. Maximum Mutation Reinforcement Learning for Scalable Control. *ArXiv preprint* abs/2007.13690 (2020). https://arxiv.org/abs/2007.13690
[95] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement learning: An introduction*. MIT Press.
[96] Yunhao Tang. 2021. Guiding Evolutionary Strategies with Off-Policy Actor-Critic. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. 1317–1325.
[97] Yunhao Tang and Krzysztof Choromanski. 2020. Online hyper-parameter tuning in off-policy learning via evolutionary strategies. *ArXiv preprint* abs/2006.07554 (2020). https://arxiv.org/abs/2006.07554
[98] Rohan Tangri, Danilo P. Mandic, and Anthony G. Constantinides. 2022. Pearl: Parallel Evolutionary and Reinforcement Learning Library. *ArXiv preprint* abs/2201.09568 (2022). https://arxiv.org/abs/2201.09568
[99] Bryon Tjanaka, Matthew C Fontaine, Julian Togelius, and Stefanos Nikolaidis. 2022. Approximating Gradients for Differentiable Quality Diversity in Reinforcement Learning. *ArXiv preprint* abs/2202.03666 (2022). https://arxiv.org/abs/2202.03666
[100] Graham Todd, Madhavun Candadai, and Eduardo J. Izquierdo. 2020. Interaction between evolution and learning in nk fitness landscapes. In *Artificial Life Conference Proceedings*. MIT Press, 761–767.
[101] Tingwu Wang and Jimmy Ba. 2019. Exploring model-based planning with policy networks. *arXiv preprint arXiv:1906.08649* (2019).
[102] Yuxing Wang, Tiantian Zhang, Yongzhe Chang, Bin Liang, Xueqian Wang, and Bo Yuan. 2022. A Surrogate-Assisted Controller for Expensive Evolutionary Reinforcement Learning. *ArXiv preprint* abs/2201.00129 (2022). https://arxiv.org/abs/2201.00129
[103] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Rémi Munos, Koray Kavukcuoglu, and Nando de Freitas. 2017. Sample Efficient Actor-Critic with Experience Replay. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=HyM25Mqel
[104] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3 (1992), 279–292.
[105] Bruce H. Weber and David J. Depew. 2003. *Evolution and learning: The Baldwin effect reconsidered*. Mit Press.
[106] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. 2014. Natural evolution strategies. *The Journal of Machine Learning Research* 15, 1 (2014), 949–980.
[107] Jiaxing Zhang, Hoang Tran, and Guannan Zhang. 2020. Accelerating Reinforcement Learning with a Directional-Gaussian-Smoothing Evolution Strategy. *ArXiv preprint* abs/2002.09077 (2020). https://arxiv.org/abs/2002.09077
[108] Han Zheng, Pengfei Wei, Jing Jiang, Guodong Long, Qinghua Lu, and Chengqi Zhang. 2020. Cooperative Heterogeneous Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/ca3a9be77f7e88708afb20c8cdf44b60-Abstract.html