Rapid Phenotypic Landscape Exploration through Hierarchical Spatial Partitioning

TOKARCHUK, LN; Smith, D; Wiggins, G; 14th International Conference on Parallel Problem Solving from Nature

Rapid Phenotypic Landscape Exploration through Hierarchical Spatial Partitioning

Davy Smith, Laurissa Tokarchuk and Geraint Wiggins

Queen Mary University of London,
School of Electronic Engineering and Computer Science,
London, E1 4NS
{david.smith,laurissa.tokarchuk,geraint.wiggins}@qmul.ac.uk

Abstract Exploration of the search space through the optimisation of phenotypic diversity is of increasing interest within the field of evolutionary robotics. Novelty search and the more recent MAP-Elites are two state of the art evolutionary algorithms which diversify low dimensional phenotypic traits for divergent exploration. In this paper we introduce a novel alternative for rapid divergent search of the feature space. Unlike previous phenotypic search procedures, our proposed Spatial, Hierarchical, Illuminated Neuro-Evolution (SHINE) algorithm utilises a tree structure for the maintenance and selection of potential candidates. SHINE penalises previous solutions in more crowded areas of the landscape. Our experimental results show that SHINE significantly outperforms novelty search and MAP-Elites in both performance and exploration. We conclude that the SHINE algorithm is a viable method for rapid divergent search of low dimensional, phenotypic landscapes.

Keywords: algorithm design, phenotypic diversity, neuroevolution, evolutionary robotics

1 Introduction

Divergent evolutionary search methods are receiving increasing interest in the evolutionary robotics community. Optimising phenotypic diversity within a population has been shown to avoid convergence towards local optima [5], to provide diverse ranges of solutions in a given domain, [4, 7, 8] and to assist with the adaptability of robot controllers [2]. Novelty search, introduced in [5] and the more recent multi-dimensional archive of phenotypic elites (MAP-Elites) [10], are two algorithms which utilise divergent phenotypic search. In this paper we introduce the Spatial, Hierarchical, Illuminated Neuro-Evolution (SHINE) algorithm, a novel method which the authors show explores low dimensional phenotypic landscapes more thoroughly and rapidly than the current state of the art. Similarly to MAP-Elites, our proposed SHINE algorithm selects future populations from an archive of previous solutions. However, the archive in the SHINE algorithm is maintained within an hierarchical, spatially partitioned tree structure. Both the weighting of offspring selection and the number of representatives assigned

to the archive are calculated from the depth of the vertices within which the solutions reside. Candidate solutions which exhibit phenotypic traits in more crowded areas of the landscape are assigned to vertices deeper within the tree, and are penalised accordingly. This allows the evolutionary trajectory to focus on larger, shallower areas of the landscape, producing a divergent, and iteratively more focused search procedure.

This paper is organised as follows. In section 2 we give a brief overview of the use of divergent phenotypic search within evolutionary robotics. In section 3, we introduce our proposed SHINE algorithm, highlighting the methods for archive management, spatial partitioning and selection of offspring in a 2-dimensional, quadtree implementation. An initial experimental domain, selected to assess the ability of the SHINE algorithm to explore the phenotypic landscape, is presented in section 4. Our results, which are presented in section 5, highlight that SHINE significantly outperforms both novelty search and MAP-Elites. In section 6 we conclude that the hierarchical procedure adopted by the SHINE algorithm is a promising method for rapid divergent phenotypic search.

2 Related Work

Novelty Search. Novelty search, as proposed by Lehman and Stanley [5], is an algorithm which removes the need for a traditional objective function through the assignment of high fitness values to novel behaviours in a population. The objective fitness function is replaced by a behavioural distance metric, which is used to determine the novelty of an individual in a population. High novelty is assigned to individuals which exhibit features with a large distance to both the rest of the population and an archive of previously encountered, highly novel phenotypic traits.

Although novelty search has been shown to outperform objective fitness search, especially in deceptive domains, it has been shown that the assessment of behavioural novelty alone is insufficient as a generalisable evolutionary technique in many tasks, especially in domains with large feature spaces [1, 9].

MAP-Elites. More recently, the MAP-Elites algorithm, as introduced in [2, 10] is an evolutionary procedure that aims to find the highest performing solution at each point in a low dimensional behaviour space. It is a hybridization of objective driven and divergent search. In MAP-Elites, evolution proceeds through the maintenance of an archive of previously high performing individuals, with each individual being assigned to bin within a discrete, low dimensional representation of the feature space. Offspring for subsequent generations are randomly selected from the archive of high performing, yet phenotypically diverse individuals.

Due to the ability of MAP-Elites to highlight the highest performing solutions in a phenotypic landscape, Mouret and Clune introduce the term *illumination algorithm* to separate it from traditional optimisation algorithms [10].

3 Spatial, Hierarchical, Illuminated Neuro-Evolution

SHINE is an illumination algorithm designed for rapid exploration of low-dimensional feature spaces. SHINE promotes divergent search through penalising solutions which are in more crowded areas of a predefined, low dimensional phenotypic landscape. The algorithm utilises a spatially partitioned tree for the maintenance of an archive of phenotypic representatives. The mechanisms applied to both the storage and selection of the representatives are designed specifically to weight subsequent generations towards more offspring in sparse areas of the landscape.

The SHINE algorithm shares similarities to both novelty search and MAP-Elites. As in MAP-Elites, SHINE maintains an archive of previous solutions which are selected for inclusion by low-dimensional discrete phenotypic traits. However, SHINE utilises an hierarchical, spatially partitioned tree structure for archive maintenance. MAP-Elites stores a single elite within each area of the feature space; the current best performing individual at an objective function. SHINE maintains multiple individuals within each vertex of the archive tree which are chosen by their distance to the boundaries of their particular phenotypic trait, in a manner more aligned with novelty search. Therefore, the SHINE algorithm also differs from MAP-Elites in that it directly aims to optimise sparse areas of the feature space. Here we introduce the main SHINE procedure, outlining a 2-dimensional implementation which utilises a quadtree structure [11].

3.1 The Algorithm

The main procedure of the SHINE algorithm, (Algorithm 1) begins by initializing a random population P with n random individuals (Lines 1-5). In each generation, every individual ρ is assessed in the domain and a phenotypic descriptor is measured and assigned to μ (lines 7-9). The tree, $\mathcal{T}$, is queried with the descriptor μ (line 9). After all individuals in the current population have been assessed and the tree structure updated, P is added to the archive (line 11). A new archive is calculated and assigned to $\mathcal{X}$ (line 12). All individuals are removed from the population, which is then repopulated with mutated offspring from the updated archive $\mathcal{X}$ via weighted roulette selection (lines 14-18). This procedure is repeated until a terminating condition is met, or alternatively after a predefined number of generations (line 19).

Phenotypic Tree. In a similar manner to MAP-Elites, the SHINE algorithm progresses through the maintenance of an archive of genomes which are selected for inclusion by a measured phenotypic trait. However, SHINE maintains an archive of potential genomes in an hierarchical, spatially partitioned tree.

The number of dimensions and the bounding volume of the phenotypic descriptor are required to initialise the root vertex of the phenotypic tree. In this paper, we focus upon the 2-dimensional implementation of the algorithm, resulting in a quadtree structure [11]. We define a phenotypic descriptor as an ordered pair

Algorithm 1 Main SHINE procedure

Require: α: max tree depth, β: vertex division level, $\mathcal{V}$: phenotypic tree
1: **procedure** SHINE
2: $P \leftarrow \varnothing$
3: **while** $|P| < n$ **do**
4: $P \leftarrow$ RandomIndividual()
5: **end while**
6: **do**
7: **for** $\rho \in P$ **do**
8: $\mu \leftarrow$ PerformTrial(ρ)
9: QueryTree(μ, $\mathcal{V}$)
10: **end for**
11: UpdateArchive(P, $\mathcal{V}$)
12: $\mathcal{X} \leftarrow$ CurrentRepresentatives($\mathcal{V}$)
13: $P \leftarrow \varnothing$
14: **while** $|P| < n$ **do**
15: $x \leftarrow$ RouletteSelection($\mathcal{X}$)
16: $x' \leftarrow$ Mutate(x)
17: $P \leftarrow P \bigcup x'$
18: **end while**
19: **while** Terminate() is false
20: **end procedure**

$\mu = (x, y)$. However, the algorithm may be extended to phenotypic descriptors with higher numbers of dimensions. Let $|\mu|$ represent the number of dimensions of a phenotypic descriptor and let $c = 2^{|\mu|}$. Each vertex will be subdivided into c child vertices (each dimension being split into 2 equal regions). Therefore, 3-dimensional traits ($|\mu| = 3$) would require an octree ($c = 2^3$) structure.

The SHINE algorithm requires 2 pre-defined constants to control the subdivision of the tree. We define constant α to be the maximium depth of the tree and β as the maximum number of points which may fall within a leaf vertex before it is divided. These constants are used to determine both the underlying phenotypic tree structure and the archive of representatives.

A series of trial runs in our experimental domain were performed with a range of α and β values: $\alpha = (3, 4, 5, ..., 12, 13, 14)$, $\beta = (20, 40, 60, ..., 120, 140, 160)$. The values $\alpha = 7$ and $\beta = 80$ produced the most reliable and optimal results and are therefore used in our experimental setup. Testing in further domains and with differing population sizes would be required to ascertain whether these values are universally optimal.

The QueryTree(μ, $\mathcal{V}$) method (line 9, Algorithm 1) determines the development of the tree structure. Figure 1 illustrates an example quadtree structure with parameters $\alpha = 4$ and $\beta = 2$. During each generation, all individuals are assessed and the tree is queried with their phenotypic descriptor, μ. Let v represent the relevant vertex of $\mathcal{V}$. Let the bounding area of $v = [v_{x1} : v_{x2}] \times [v_{y1} : v_{y2}]$, where $v_{x1} < \mu_x \leq v_{x2} \wedge v_{y1} < \mu_y \leq v_{y2}$. Let v_d be the depth within the tree and $|v|$ be the number of descriptors currently assigned to v. If the capacity

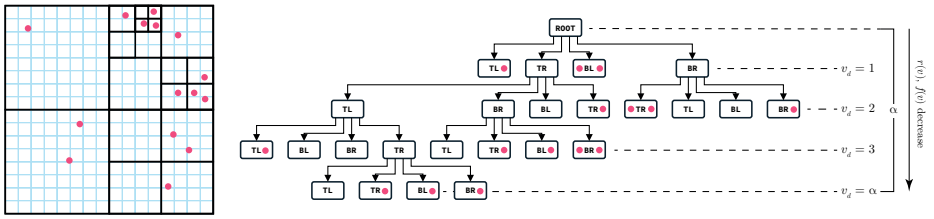


Fig. 1: An overview of spatial partitioning in the SHINE archive. ($\alpha = 4, \beta = 2$).

of v has been exceeded and the maximum depth has not been reached, such that $|v| > \beta \wedge v_d < \alpha$, then v is subdivided into 4 equal sized regions, i. e., top-left, top-right, bottom-left and bottom-right (TL, TR, BL, BR, figure 1). All descriptors within v are then assigned to their relevant child vertices.

Archive Management. After the tree has been queried by the population, the resulting structure is utilised to determine the distribution of the archive of representatives from which subsequent populations are selected. Membership of the archive is weighted dependant upon the depth of the representatives' containing vertex. Shallower vertices in the tree structure are assigned more representatives. Representatives do not alter the structure of the tree, rather the relevant vertex for a potential representative's phenotypic descriptor determines whether it is added to the archive. Let $|\mu|$ represent the dimensions of a phenotypic descriptor and let $c = 2^{|\mu|}$. Equation (1) defines the maximum number of representatives $r(v)$ which may be assigned to a particular vertex.

$$r(v) = (v_d - \alpha + 1)^c \tag{1}$$

The number of representatives within a single vertex will therefore fall within the range $1 \leq r(v) \leq (\alpha + 1)^c$. Let $\mathcal{X}_v$ be the set of all representative within a vertex, v. If the capacity of v is reached, such that $|\mathcal{X}_v| = r(v)$, representatives from $\mathcal{X}_v$ are selected for addition or removal based upon a distance function $d(x)$. This distance function determines the distribution of representatives *within a single leaf vertex*. In alignment with this, let x be a potential representative for inclusion within the archive, where $x \notin \mathcal{X}_v$. Let $w \in \mathcal{X}$ be the weakest current representative $w = \underset{\forall i \in \mathcal{X}_v}{\arg\max}\ d(i)$. The updated archive of representatives, which we define as $\mathcal{X}_v^{'}$, is determined as in equation (2).

$$\mathcal{X}_v^{'} = \begin{cases} \mathcal{X}_v \bigcup x & \text{if } |\mathcal{X}_v| < r(v) \\ \mathcal{X}_v & \text{if } |\mathcal{X}_v| = r(v) \text{ and } d(x) > d(w) \\ \{\mathcal{X}_v \setminus w\} \bigcup x & \text{if } |\mathcal{X}_v| = r(v) \text{ and } d(x) \leq d(w) \end{cases} \tag{2}$$

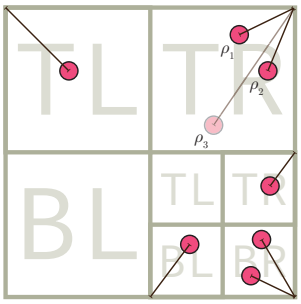


Fig. 2: An illustration of the *corner* sorting method for representative selection.

Dependant upon the particular type of search required, various metrics may be proposed. For example, defining $d(x)$ as an objective function would allow the archive to behave in a similar manner to the MAP-Elites algorithm [10], selecting elite representatives for inclusion within the phenotypic tree. We also suggest that metrics based upon novelty search [5] or hybrid novelty-objective measures [12] may be of particular interest for further testing of the algorithm in different domains.

In our experiment, presented in section 4, we utilise the *corner* distance metric, a function which favours representatives in the outer corners of the containing vertex, encouraging representatives to focus on the areas closest to neighbouring vertices and increasing the chance of mutated offspring to acquire phenotypic traits in neighbouring cells. Figure 2 illustrates our *corner* method for representative selection. Representatives are sorted by distance from the outer corner of their assigned vertex's position in the quad tree structure (i.e. representatives in top-left vertices are sorted by their distance from top left corner of the vertex). Once the number of representatives exceeds the maximal threshold, as defined in equation (1), the representative with the largest distance is removed.

Proportional Selection. SHINE utilises a traditional roulette wheel method for the selection of offspring. Potential solutions are selected from the complete set of current representatives within the tree $\mathcal{X} = \{\mathcal{X}_{v_1} \bigcup, ..., \bigcup \mathcal{X}_{v_{|\mathcal{V}|}}\}$. The fitness $f(x)$ of a representative x in vertex v is obtained by calculating the reciprocal of the sum of the vertices' depth v_d and its normalised population $\frac{v_p}{\beta}$. Defined as $1/(v_d + \frac{v_p}{\beta})$ and simplified in equation (3).

$$f(x) = \frac{\beta}{\beta v_d + v_p} \tag{3}$$

This fitness assignment results in a lower probability of selection of representatives within smaller (deeper within the tree) and more crowded (higher population) areas of the phenotypic landscape, allowing the search procedure to concentrate on larger and sparser vertices within the tree.

4 Experimental Evaluation

Domain. The aim of our experiment is to assess the diversity and thoroughness of phenotypic exploration in an evolutionary trajectory optimised with the SHINE algorithm in comparison to novelty search and MAP-Elites. Therefore, we select a domain with a deceptive objective function and which requires a high level of exploration to produce a successful solution.

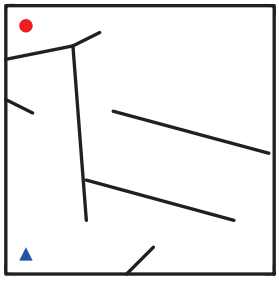


Fig. 3: The HARD maze domain. Triangle indicates agent start position, circle indicates exit.

Our experimental domain is taken directly from previous studies which have assessed novelty search and variants of the algorithm [3,5,6]. The maze used in our experiment, the HARD maze, is classified as a deceptive domain, particularly difficult for objective algorithms to reliably find solutions (Figure 3). The maze is of the size 1000×1000 units, the agent has a size of 20 units and successfully reaching the exit requires the agent to be within 20 units. Each agent is given 4000 time steps to complete the maze. Populations of 200 controllers were optimised for 1000 generations. The agent controllers are neural networks which are evolved using the NEAT algorithm [13], with the speciation mechanism deactivated. As in [3, 5, 6], the objective fitness of a solution ρ is calculated as $f(\rho) = l - dist(\rho, e)$, where l is the diagonal length of the maze and e is the exit to the maze. The phenotypic descriptor is calculated from the ending position of the agent, $\mu = (\rho_x, \rho_y)$.

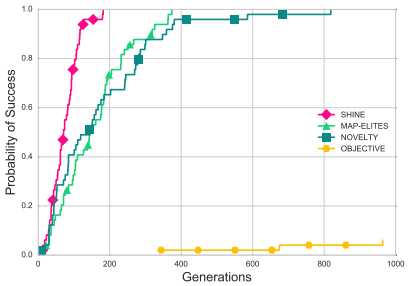
We assess 4 algorithms in our experiment — traditional objective based search (OBJECTIVE), novelty search (NOVELTY), MAP-Elites (MAP-ELITES), and our proposed SHINE algorithm (SHINE). The algorithms were repeated in each domain 50 times with a different random seed in each trial. In order to ensure consistency between algorithms, identical random seed values were given to each of the algorithms in each trial. The performance of each algorithm was determined by the number of generations taken to locate the exit in the domain.

The simulation was performed using a bespoke domain written in the C++ programming language, developed to be similar to the original maze domain experiments in [5, 6]. The implementation of the NEAT algorithm used was developed as an extension to the MultiNEAT software in the C++ language[1].

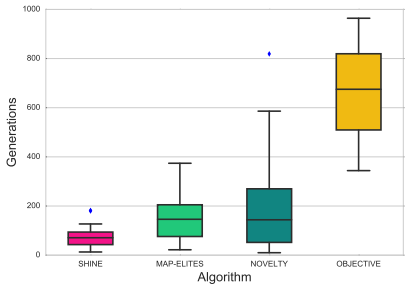
Domain Coverage. The cumulative coverage of the domain is calculated at each generation in the trial over 1000 generations. The domain is divided into a 2-dimensional matrix M, where $|M| = n \times n$. In our presented results, $n = 30$. The final position of an individual (ρ_x, ρ_y) is mapped to the corresponding region of M. Let M' be the set of the regions of M which contain individuals: $M' = \{x : x \in M \wedge |x| > 0\}$. Domain coverage is then calculated as $\frac{|M'|}{|M|}$.

Exploration Uniformity. The spread of the population is measured through the calculation of *exploration uniformity* in a similar manner to [3]. To ascertain the speed at which exploration occurs for each algorithm, values are calculated at each generation in the trial rather than cumulatively over the whole trial as in [3]. Again, the population is mapped to the discrete matrix M. Let P_t be the set of

[1] © 2012 Peter Chervenski. http://multineat.com/index.html



(a) Probability of success in HARD maze domain



(b) Number of Generations taken to locate the exit in the HARD maze domain in successful trials[2].

Fig. 4: Performance results from HARD maze domain

individuals in the population at generation t and let Ψ_t be the distribution of P_t over M. The exploration uniformity of the population, $D(P_t)$, is calculated as the similarity between Ψ_t and the uniform distribution U. As in [3] the distance metric used is the *Jensen-Shannon* distance (JSD). The exploration uniformity at generation t is thus defined as:

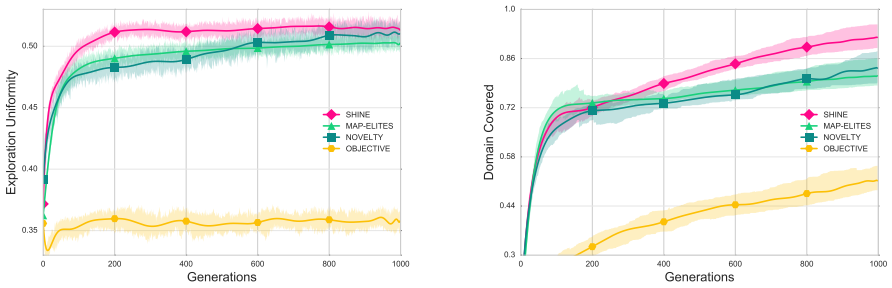
$$
\begin{aligned}
&D(P_t) = 1 - JSD(\Psi_t, U),\ \textit{where}: \\
&\Psi_t = \left(\frac{|I_1|}{|P_t|}, ..., \frac{|I_{|P_t|}|}{|P_t|} \right), I_r = \{ i \in P_t : region(i) = r \} \\
&U = \left(\overbrace{\frac{1}{|M|} \times \cdots \times \frac{1}{|M|}}^{n^2 \text{ times}} \right)
\end{aligned}
\tag{4}
$$

5 Results

Performance. As illustrated in figure 4a, all 3 algorithms located solutions to the maze in all 50 trials, resulting in a probability of success of 1.0. Maximum probability of success is reached significantly faster ($p < 0.001$) by the SHINE algorithm, after 182 generations, compared with 374 generations for MAP-ELITES and 819 generations for NOVELTY. Both NOVELTY and MAP-ELITES follow a similar gradient of ascent, however NOVELTY requires a higher number of generations to locate a solution in 3 of the trials.

Figure 4b shows the number of generations taken to find a successful solution. The SHINE algorithm requires a significantly fewer number of generations, with a median value of 71. MAP-ELITES and NOVELTY achieve similar results, with median values of 146 and 141 generations respectively.

[2] SHINE, NOVELTY and MAP-ELITES were successful in all trials.



(a) Exploration uniformity of the current population at each generation.

(b) Cumulative proportion of domain coverage by the population after each generation.

Fig. 5: Diversity of the algorithms within the HARD domain. (Shaded area indicates 25^{th} to 75^{th} percentiles.)

Diversity Figure 5a shows the exploration uniformity for each of the algorithms over 1000 generations. The maximum mean level of exploration uniformity is achieved by the SHINE algorithm, 0.51912 after 772 generations. However, it achieves comparably high levels after 232 generations, remaining relatively stable throughout the evolution. Both MAP-ELITES and NOVELTY fail to achieve this maximal level within 1000 generations, however the exploration uniformity is still increasing for both algorithms at the end of the trial. The maximum mean level achieved by MAP-ELITES is 0.50584 after 984 generations. NOVELTY achieves a maximal value of 0.51408 after 988 generations. Therefore an evolutionary run with a higher number of generations may allow MAP-ELITES and NOVELTY to achieve a level of exploration uniformity similar to SHINE. Figure 5b shows the proportion of the domain covered by the population. All three algorithms produce similar levels of domain coverage for the initial 400 generations. However, beyond this SHINE covers significantly more of the domain than both NOVELTY and MAP-ELITES.

6 Conclusion

In this paper we have introduced a novel method for rapid exploration of low dimensional feature spaces. Our experimental evaluation in a deceptive simulated maze domain shows that the SHINE algorithm outperforms both novelty search and MAP-Elites, two state of the art algorithms for divergent phenotypic search. We have shown that the hierarchical tree structure and approach taken for archive maintenance and offspring selection in the SHINE algorithm are viable methods for rapid phenotypic exploration.

Further experimental validation is required in order to establish the performance of the SHINE algorithm in domains with a less direct mapping between

the feature space and the objective landscape. The authors suggest that a replacement of the *corner* method presented in this paper to an objective function would allow SHINE to be compared more directly with MAP-Elites in objective focussed domains. The authors are aware of the limitations in testing within a simulated environment. MAP-Elites has been shown to be extendible to the real world application of robot controllers [2]. Therefore we suggest a future direction to be the assessment of SHINE beyond simulation, in real world domains.

Acknowledgements. This work was funded by EPSRC through the Media and Arts Technology Programme, an RCUK Doctoral Training Centre EP/G03723X/1. Computational facilities were provided by the MidPlus Regional Centre of Excellence for Computational Science, Engineering and Mathematics, under EPSRC grant EP/K000128/1.

References

1. G. Cuccu and F. Gomez. When novelty is not enough. In *Applications of Evolutionary Computation*, pages 234–243. Springer, 2011.
2. A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.
3. J. Gomes, P. Mariano, and A. L. Christensen. Devising effective novelty search algorithms: A comprehensive empirical study. In *Proceedings of the 2015 Genetic and Evolutionary Computation Conference*, pages 943–950. ACM, 2015.
4. F. J. Gomez. Sustaining diversity using behavioral information distance. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 113–120. ACM, 2009.
5. J. Lehman and K. O. Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336, 2008.
6. J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011.
7. J. Lehman and K. O. Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218. ACM, 2011.
8. J. Lehman, K. O. Stanley, and R. Miikkulainen. Effective diversity maintenance in deceptive domains. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, pages 215–222. ACM, 2013.
9. J.-B. Mouret. Novelty-based multiobjectivization. In *New horizons in evolutionary robotics*, pages 139–154. Springer, 2011.
10. J.-B. Mouret and J. Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.
11. H. Samet. The quadtree and related hierarchical data structures. *ACM Computing Surveys (CSUR)*, 16(2):187–260, 1984.
12. D. Smith, L. Tokarchuk, and G. Wiggins. Exploring conflicting objectives with madns: Multiple assessment directed novelty search. In *Companion Proceedings of the 2016 Genetic and Evolutionary Computation Conference*. ACM, 2016.
13. K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.