

Coupling Novelty and Surprise for Evolutionary Divergence

Daniele Gravina
Institute of Digital Games
University of Malta
daniele.gravina@um.edu.mt

Antonios Liapis
Institute of Digital Games
University of Malta
antonios.liapis@um.edu.mt

Georgios N. Yannakakis
Institute of Digital Games
University of Malta
georgios.yannakakis@um.edu.mt

ABSTRACT

Divergent search techniques applied to evolutionary computation, such as novelty search and surprise search, have demonstrated their efficacy in highly deceptive problems compared to traditional objective-based fitness evolutionary processes. While novelty search rewards unseen solutions, surprise search rewards unexpected solutions. As a result these two algorithms perform a different form of search since an expected solution can be novel while an already seen solution can be surprising. As novelty and surprise search have already shown much promise individually, the hypothesis is that an evolutionary process that rewards both novel and surprising solutions will be able to handle deception in a better fashion and lead to more successful solutions faster. In this paper we introduce an algorithm that realises both novelty and surprise search and we compare it against the two algorithms that compose it in a number of robot navigation tasks. The key findings of this paper suggest that coupling novelty and surprise is advantageous compared to each search approach on its own. The introduced algorithm breaks new ground in divergent search as it outperforms both novelty and surprise in terms of efficiency and robustness, and it explores the behavioural space more extensively.

CCS CONCEPTS

•Computing methodologies → Search methodologies; Genetic algorithms; Evolutionary robotics;

KEYWORDS

Surprise search; novelty search; divergent search; deception; fitness-based evolution; map navigation; NEAT

ACM Reference format:

Daniele Gravina, Antonios Liapis, and Georgios N. Yannakakis. 2017. Coupling Novelty and Surprise for Evolutionary Divergence. In *Proceedings of GECCO '17, Berlin, Germany, July 15-19, 2017*, 8 pages. DOI: <http://dx.doi.org/10.1145/3071178.3071179>

1 INTRODUCTION

One of the most critical aspects of optimisation is the design of the objective function — or in the context of evolutionary computation (EC) the *fitness function* [10]. The fitness function determines how

the evolutionary algorithm will explore the search space by assessing the goodness of the evolved solutions. The widely accepted approach is to design a fitness function that explicitly rewards solutions in terms of their goodness towards the objective of the search. Recent findings, however, in the area of divergent evolutionary search [15, 17, 20, 24] suggest that rewarding intrinsic properties of the search can overcome *deception* [10, 29] which is widely considered a direct measure of problem hardness and a major challenge within EC. The deception of a problem is directly linked to the fitness landscape: the challenge is to design a fitness function capable of guiding search away from local optima, which can, in turn, make the search hard for an objective-based fitness function. Further, an ill-posed fitness function can be detrimental for the search as rewarding local optima drives the search away from the global solution. To overcome such challenges divergent search — instead of an explicit objective — rewards the degree of an individual's divergence such as the degree of *novelty* [17] or *surprise* [15].

Novelty and surprise have been used as core notions in evolutionary divergent search; both strategies have proven to be more successful than objective search in highly deceptive tasks such as in robot maze navigation [15, 17] and in robot locomotion [18]. Novelty search (NS) is built on the principle of open-ended evolution within artificial life [3] which does not consider explicit objectives; instead, the dominant model is open-ended search [1, 31]. Similarly surprise search (SS) is built on the divergent search paradigm and draws from literature in cognitive science and computational creativity suggesting that not only humans are capable of self-surprise but, most importantly, that surprise is a core internal driver of creativity and its final outcomes [12]. It has been empirically found that a human-centric cognitive process as such can help computers solve difficult problems [15, 32]. Deceptive problems are in particular need of such divergent search processes — as shown from earlier studies [15, 17, 20, 24].

Both novelty and surprise search are algorithms for divergent search which ignore the problem's objectives. However, each algorithm rewards solutions differently: NS rewards solutions which exhibit dissimilar behaviour from those in the current and in previous populations, while SS rewards solutions which diverge from expected behaviours based on past trends. These rewards are orthogonal, since a solution can be novel without being surprising (i.e. this divergence from past solutions is expected), and vice versa if a solution breaks expectations by revisiting explored areas of the search space (i.e. the solution is not novel). Our hypothesis in this paper is that by rewarding both novel and surprising solutions to different degrees, a new type of more sophisticated search can emerge. We argue that by coupling the search for unseen (NS) with the search for unexpected (SS) solutions we will end up with an algorithm that explores unseen points in the search space and at the same time also rewards deviations from predicted search trends.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

GECCO '17, Berlin, Germany

© 2017 ACM. 978-1-4503-4920-8/17/07...\$15.00

DOI: <http://dx.doi.org/10.1145/3071178.3071179>

Combining the two concepts (novelty and surprise) into a single reward mechanism will provide a more nuanced evaluation of divergence and can thus lead evolution to better handle deceptive problems.

To test our hypothesis, we introduce a simple evolutionary algorithm we name novelty-surprise search (NSS) that sums and rewards both the degrees of novelty and surprise of an evolved solution. There are several ways of combining novelty and surprise as concepts, including multi-objective approaches [7], alternating between the two rewards, or merging their deviation formulas. As the first attempt at a combined novelty-surprise search algorithm, this paper uses the simplest alternative of aggregating the novelty score and the surprise score into a single reward. Given that the two concepts are in theory orthogonal, we assume this aggregated approach should be sufficiently efficient. Future work can use the results of NSS as a baseline to test improvements of this simple, yet fast and straightforward method.

In order to evaluate our hypothesis for NSS, we adopt the methodology and the testbed proposed in [17] and used in [15] to evolve robot controllers via neuroevolution of augmenting topologies (NEAT) [27] for three maze navigation tasks. We compare the performance of NSS against novelty search and surprise search via two performance measures: efficiency (i.e. maximum fitness obtained) and robustness (i.e. the number of times a problem is solved). Our results show that coupling surprise and novelty improves algorithm performance more than its constituent parts, and proves advantageous with regards to efficiency and robustness.

2 BACKGROUND

The concept of *deception* in the context of evolutionary computation was introduced by Goldberg [9], as a way to describe instances where highly-fit blocks, when recombined, can lead the search away from the global optimum. Since then, the notion of deception has been refined and expanded, and it is used to describe problems that challenge the evolutionary search for a solution. Deception, sampling error [23] and a rugged fitness landscape [16] are often considered as responsible for EC-hardness; the fitness landscapes, especially in the case of combinatorial optimisation problems, can affect the optimisation process when performing neighbourhood search [26]. This search process assumes there is a high correlation between the fitness of neighbouring points and that genes in the chromosome are independent of each other. Genes' independence refers to the concept of *epistasis* [5], which is considered one of the factors of EC-hardness. When too many genes are influenced by other genes in the chromosome, the epistasis is high and the algorithm searches for a unique optimal combination of those genes without any improvement in terms of global fitness [5].

A deceptive landscape, therefore, actively draws away the search from the global optimum and traps the algorithm in some local optima. Several approaches have been proposed to counter this behaviour, as surveyed in [20]. Speciation [27] and niching [30], for example, have been proposed as diversity maintenance approaches as they enforce local competition among similar solutions. Another approach is *coevolution*, where the fitness is computed based on a competition between individuals in the same population [2]. This should ideally lead to an arms race towards a better gradient for

search, but it runs the risk of obtaining only mediocre solutions, as it can happen that either all competitors find a poor solution or one competitor performs so well that the other competitors cannot improve [8].

2.1 Novelty search

Novelty search [17] is a recent alternative to evolutionary search that explicitly ignores the objective of the problem it attempts to solve. This search method selects individuals based on how diverse they are with respect to the solutions found so far. Novelty search therefore attempts to reward novel behaviours in the current population and in past generations, by keeping a *novelty archive* of past novel behaviours. Novelty search is neither a random nor an exhaustive search, as it learns to explore uncharted areas of the search space [28]. In the general formulation of novelty search, each individual (i) is evaluated based on the distance (d_n) from the n nearest neighbours in both the population and the archive:

$$n(i) = \frac{1}{n} \sum_{j=0}^n d_n(i, j), \quad (1)$$

where j is the j -th nearest neighbour with respect to novelty distance d_n and $n(i)$ is the novelty score for the individual i . Neighbors are selected from the current population and from the novelty archive.

Novelty search has demonstrated its effectiveness in several domains such as maze navigation and robot control [17], generation of images [19] and generation of game content [21, 22].

2.2 Surprise search

Surprise search [32] is an alternative approach to divergent search which rewards *unexpected* rather than *unseen* solutions in the search space. This is accomplished via a model that predicts where search will be in the next generation based on the behavioural history of evolution. The derived predictions of the model are used to reward deviations from the expected [13, 32]. Surprise search attempts to mimic a self-surprise process [11] which explores the prediction space instead of the actual behavioural space. It is important to note that the prediction space can be different from the behavioural space. For instance, in the maze navigation problem a prediction can be an unreachable position of the maze.

Surprise search is composed by two main phases: the prediction model (Eq. 2) and the deviation formula (Eq. 3) phases. First the algorithm predicts a number of future behaviours (\mathbf{p}) based on a selected number of past behaviours (as determined by the history parameter h) and on the locality of behavioural information (expressed by the k parameter) which are modelled through m :

$$\mathbf{p} = m(h, k). \quad (2)$$

Once the predicted behaviours are in place the algorithm rewards the members of the current population based on the distance of the n closest predicted behaviours of the current generation as follows:

$$s(i) = \frac{1}{n} \sum_{j=0}^n d_s(i, p_{i,j}), \quad (3)$$

where $s(i)$ is the surprise score of individual i that is computed as the average distance (d_s) of i from its n closest predictions ($p_{i,j}$).

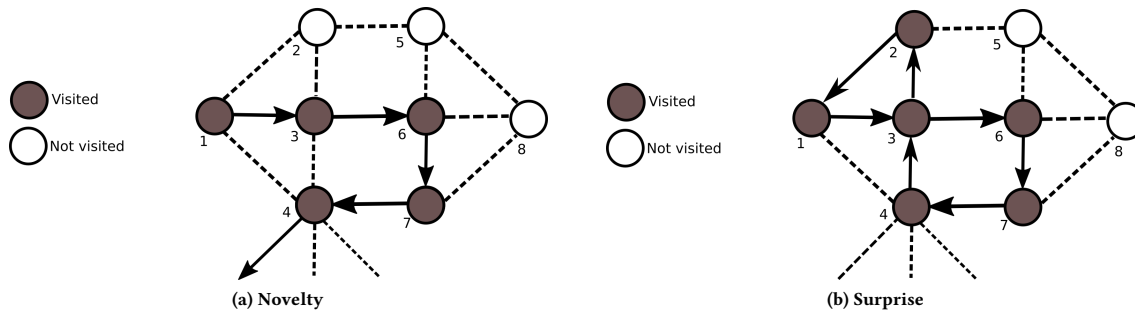


Figure 1: Graph model example: an agent is visiting a graph made of nodes connected in a fixed configuration. Every step, an agent follows a particular model to decide which node to visit next. In (a) the agent visit only unvisited nodes (to maximise the novelty score), while in (b) the agent is trying to maximise the surprise reward, by deviating from the predictions made with the prediction model m .

More details about surprise search can be found in [15, 32]. Surprise search was shown to outperform objective search in a deceptive environment and to be more robust compared to novelty search [15]. Further, a constrained version of surprise search has been highly effective for game weapon generation tasks [14]. The generated weapons have guaranteed high quality – imposed by weapon balance and weapon effectiveness constraints – and are characterised by high diversity (achieved via the reward of surprise), thereby, directly realising *quality diversity* [25].

3 COUPLING NOVELTY AND SURPRISE

Earlier work [15, 17] has shown that objectives tend to be counter-productive when searching for the global solution in deceptive problems as they tend to mislead the search and often penalize potentially critical stepping stones of evolution. Divergent search algorithms such as novelty or surprise search, on the other hand, have shown promise in highly deceptive problems even though recent findings have demonstrated that some deceptive problems can be challenging even for divergent search [4].

The difference between novelty and surprise can be exemplified by considering an agent travelling a graph made of nodes connected in a fixed configuration, starting from the node labelled as 1. Every step the agent has to decide which node to visit next, and based on a certain model it will get a reward based on the decision made. The objective of the agent is to maximise the immediate reward of every step, while the final objective is to visit as many different nodes as possible. We set up two different models, where respectively we reward the agent when it visits a novel node i (novelty from Eq. (1)) or an unexpected node i (surprise from Eq. (3)). In the first case, an agent would probably try to maximise novelty by looking for novel discovery in every step it takes in the graph: this can easily lead to a situation as in Fig. 1a. In this case, the sequence of discoveries is $1 \rightarrow 3 \rightarrow 6 \rightarrow 7$: in this configuration, the agent has to decide to visit either node 4 or node 8, but as they are both novel, the agent will get the same reward no matter what it decides. Therefore, if it decides to visit node 4, the agent will never visit nodes 2, 5 and 8, as backtracking (i.e. visiting already visited nodes) is highly discouraged by the novelty reward. Instead, a surprise approach would try to deviate from the patterns learned while

visiting the nodes of the graph. In this case, the surprise model has higher chance to visit nodes not visited in the previous example, as backtracking could be the result of a self-surprising behaviour. If we take the same sequence as before ($1 \rightarrow 3 \rightarrow 6 \rightarrow 7 \rightarrow 4$), surprise would encourage visiting already seen nodes (for example node 3); if its prediction model has learned to expect a new node in every step, visiting an already seen node is unexpected. Therefore, a surprising approach might lead to visit nodes not explored by novelty, thanks to backtracking. However, the drawback of this model is that a surprising agent can easily get stuck visiting the same nodes in a loop, as backtracking can cause a “circular behaviour” (e.g. it will forever visit already visited points) as pointed out in Fig. 1b. Therefore, by combining the properties of novelty and surprise models, we can see that the agent would visit the graph in a more extensive way, as a search process that maximises novelty would push for novel discoveries, while a surprise reward would help to backtrack to already visited positions, allowing the agent to discover solutions not explored by novelty alone.

As mentioned in the introduction, the working hypothesis of the paper is that we can equip computers with better search capacities if we couple algorithms with dissimilar properties and ways of operating in the search space. In this paper we combine two divergent algorithms, novelty search and surprise search, and we assume that their complementarity in search – searching for the unseen *and* searching for the unexpected – can improve an algorithm’s performance compared to merely searching for novel or surprising solutions. As the concept of novelty is orthogonal to that of surprise and surprise can be viewed as *temporal novelty* [32], combining the two seems to be an approach with great potential. The following section introduces an EC algorithm that couples novelty and surprise into *novelty-surprise search* (NSS).

3.1 Novelty-surprise search

In the field of evolutionary computation several approaches exist for simultaneously optimising several objectives. While recent literature has proposed several multiobjective evolutionary algorithms [6], the simplest solution is to linearly combine the objectives. In this work we opt for the latter approach, as adding novelty and surprise is trivial and computationally preferred while it represents

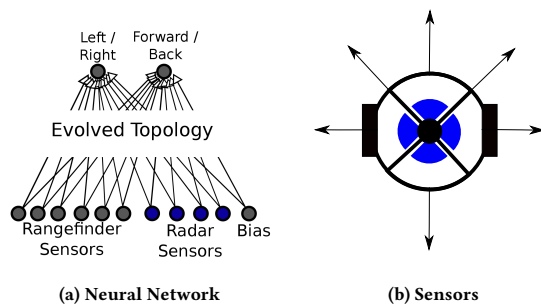


Figure 2: Robot controller for maze navigation. Fig. 2a shows the network’s inputs and outputs. Fig. 2b shows the layout of the sensors: the six black arrows are range-finder sensors, and the four blue pie-slice sensors act as a compass towards the goal.

an intermediate step towards a multiobjective implementation (discussed further in Section 6). The novelty-surprise search (NSS) algorithm executes both novelty and surprise search and at each generation it rewards an individual by adding its novelty and surprise score in the following fashion:

$$ns(i) = \lambda \cdot n(i) + (1 - \lambda) \cdot s(i), \quad (4)$$

where $ns(i)$ is the combined novelty and surprise score of individual i and $\lambda \in [0, 1]$ is a parameter that controls the relative importance of novelty versus surprise.

4 MAZE NAVIGATION TESTBED

To be able to compare our findings with earlier work in novelty and surprise search [15, 17] we test the NSS algorithm in the well-studied maze navigation task. In this task a simulated mobile robot has to find the goal in a maze in a limited number of simulation steps. The maze is designed by humans and the maze navigation is made deceptive with the presence of several dead ends, which create local optima in the search space. The robot (Fig. 2b) is controlled by an artificial neural network (Fig. 2a), with 10 inputs (6 range-finder sensors and 4 radar sensors) and 2 outputs (which control robot movement): more details can be found in [17].

Three mazes, identified as *medium*, *hard* and *very hard*, are used in this work. The first two mazes were introduced in [17] and have been studied in [15]; the very hard maze is a new maze designed by the authors. The medium maze has several dead ends in the search space (Fig. 3a), but still a straightforward path towards the goal can be found. The hard maze (see Figure 3b) is more deceptive as the path to the goal is more convoluted and goes through more distant areas to the goal than where it starts at. The very hard maze (Fig. 3c) is a modification of the hard maze with more dead ends that make the path to the goal even more complex.

The evolved robot is considered successful if it manages to find the goal within a radius of five units in 400 simulation steps for the medium and hard maze, and 500 steps for the very hard maze; the latter modification is due to the highly deceptive landscape encountered in the very hard maze, which led us to increase empirically

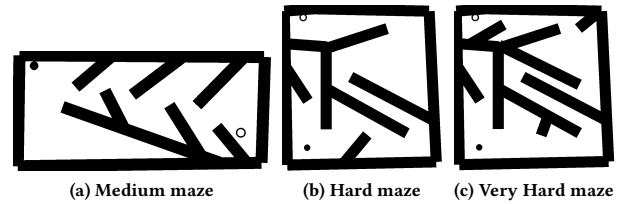


Figure 3: The robot navigation mazes that appear in [17] as “medium” and “hard”, and the new “very hard” maze. For comparative purposes the same mazes are used for all experiments in this paper. The solid and empty circle represent the robot’s starting position and the goal, respectively.

the number of steps in order to let at least one algorithm have a reasonable number of successes.

4.1 Algorithm parameters

All three algorithms use NEAT to evolve a robot controller with the same parameters as in [17]. The population size is 250 and evolution is carried out for 300 generations for the medium and hard maze and for 1000 generations for the very hard maze. The NEAT algorithm uses both speciation and recombination, as in [27].

4.1.1 Novelty Search. Novelty search uses the same novelty score and the same parameters used in [17]. While in [17] novelty is calculated as the average distance from the 15 nearest neighbours, the introduction of the new maze in this paper mandates that the n parameter of novelty search is tested empirically. For that purpose we vary n from 5 to 30 in increments of 5 across all mazes and select the n values that yield the lowest average evaluations in 50 independent runs of 300 generations for the medium and hard maze, and 1000 generations for the very hard maze. Based on the results, we select $n = 15$ for all mazes, as it obtains the minimum number of evaluations for each maze considered.

4.1.2 Surprise Search. Surprise search instead uses the surprise score described in Equation (3); as in novelty search, the behaviour of the robot is characterised by its final position in the maze. The surprise score is computed as the Euclidean distance between the robot and the two closest predicted points ($n = 2$) in this paper, whereas the prediction model is based on a one-step linear regression of two past generations ($h = 2$). Note that parameters n and h are domain dependent and are not selected empirically in this paper – they are instead selected based on the good performance achieved by surprise search in earlier studies [15]. The local behaviours used in the prediction model are obtained through k -means clustering in the behavioural space. In contrast to n and h the locality parameter k of Eq. (3) is selected empirically and based on a sensitivity analysis. The k value that yields the least evaluations, on average, in 50 independent runs of the algorithm is selected for each maze. As a result of this selection process k is 100, 50 and 200 in medium, hard and very hard maze respectively.

4.1.3 Novelty-surprise search. NSS uses the novelty and surprise scores described above and combines them linearly as in Eq. (4). The specific parameters of novelty search (n) and surprise search

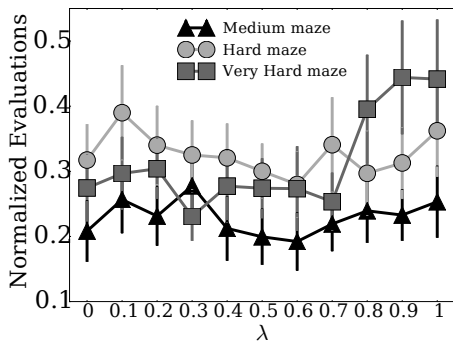


Figure 4: Sensitivity analysis: selecting λ for NSS. The figure depicts the average number of evaluations (normalized by the maximum evaluations allocated) obtained out of 50 runs (of 300 generations for the medium and hard maze, of 1000 generations for the very hard maze). Error bars represent the 95% confidence interval.

(n , h , k) remain unchanged. A core component of the proposed NSS algorithm is the λ parameter which determines the impact of both novelty and surprise in Eq. (4): higher λ values put more weight on novelty. To select appropriate λ values we run 11 experiments for each maze, each time with a different λ value ranging from 0 to 1 in increments of 0.1. Each experiment is composed of 50 runs of the NSS algorithm with the particular λ parameter.

Fig. 4 shows the average number of evaluations required to find a solution across λ , for each maze. We pick λ values that solve the corresponding maze in the fewest possible evaluations: in the medium maze and hard maze, this happens for $\lambda = 0.6$ leading to an average of $14.4 \cdot 10^3$ and $21 \cdot 10^3$ evaluations respectively. In the very hard maze the fewest evaluations on average ($57.7 \cdot 10^3$) are found when $\lambda = 0.3$.

From this analysis we can argue that in more deceptive problems, the advantage of combining novelty with surprise becomes more visible. In the medium and hard maze, the novelty score is more beneficial, while the surprise score contributes 40% to the final value for best results. Instead, in the very hard maze the surprise score is more prominent for better performance, as its contribution to the ns score through the λ parameter becomes higher. This general effect is obvious when comparing the performance of NSS against the performance of the two baselines algorithms ($\lambda = 0$ and $\lambda = 1$). A detailed comparison of NSS against novelty and surprise search is presented in the next section.

5 EXPERIMENTS

The maze navigation problem is used to compare the performance of surprise search, novelty search and NSS. The performance metrics analysed are their efficiency and robustness in all the three mazes (medium, hard and very hard maze) as proposed in [15]. All results are computed from 50 independent evolutionary runs, while the reported significance is obtained via two-tailed Mann-Whitney U-tests at a 5% significance level. It is important to note that the 50 evolutionary runs analysed here are additional to (and independent of) the ones used for parameter tuning in Section 4.1.

Table 1: Efficiency. Maximum fitness over time on average for each algorithm after a number of evaluations (E), for the three mazes considered. Values in parentheses denote the 95% confidence interval.

Maze	E	NS	SS	NSS
Medium	$25 \cdot 10^3$	293.2 (1.9)	291.0 (3.3)	294.8 (2.8)
	$50 \cdot 10^3$	295.7 (0.6)	295.8 (0.3)	295.7 (0.2)
	$75 \cdot 10^3$	295.8 (0.5)	295.9 (0.2)	295.7 (0.2)
Hard	$25 \cdot 10^3$	283.6 (4.4)	284.0 (4.7)	291.5 (3.2)
	$50 \cdot 10^3$	294.2 (1.5)	293.1 (2.8)	295.0 (1.6)
	$75 \cdot 10^3$	295.7 (0.4)	295.9 (0.2)	296.0 (0.2)
Very hard	$75 \cdot 10^3$	271.3 (3.5)	286.0 (5.1)	291.0 (3.3)
	$150 \cdot 10^3$	282.0 (4.3)	293.9 (2.6)	294.8 (1.7)
	$250 \cdot 10^3$	292.4 (2.8)	296.2 (0.2)	296.2 (0.2)

5.1 Efficiency

As in the analysis in [15, 17], efficiency is defined as the maximum fitness over time: based on [17], fitness is defined as $300 - d(i)$ where $d(i)$ is the Euclidean distance between the final position of robot i and the goal.

Table 1 shows the fitness of the three algorithms considered, for each maze, where values are averaged across 50 runs of each algorithm and the values in parentheses represent the 95% confidence interval of the average. In the medium maze, the efficiency of the three algorithms is really similar, even if on average NSS is slower at the beginning and eventually becomes faster than novelty and surprise search, especially between 10,000 and 35,000 evaluations. At the end of 75,000 evaluations, the differences are not statistically significant ($p > 0.05$). In the hard maze, NSS outperforms both NS and SS between 25,000 and 50,000 evaluations (see Table 1). Again no statistical significance can be established for final fitness ($p > 0.05$). In the very hard maze, Table 1 shows that NSS outperforms SS in the first 150,000 evaluations, and it outperforms NS for the entire evolutionary process.

5.2 Robustness

In this section we compare the algorithms' *robustness* defined as the number of successes obtained by the algorithm across time (i.e. evaluations). Fig. 5 shows the robustness of all three algorithms for each maze, collected from 50 independent runs.

In the medium maze (Fig. 5a) NSS is able to find on average more solutions than novelty search, and is also faster on average (see Table 2). On the other hand, NSS outperforms SS only in the interval between 15,000 and 50,000 evaluations.

In the more deceptive hard maze (Fig. 5b) the benefit of combining novelty and surprise is more visible. NSS clearly outperforms novelty search for the entire evolutionary process, and it outperforms surprise search between 7,500 and 40,000 evaluations. In terms of average number of evaluations, NSS is significantly faster than novelty search and faster than surprise search (see Table 2).

In the last and most deceptive testbed, the very hard maze, NSS is significantly faster than NS and faster than SS. Fig. 5c shows how combining novelty and surprise can improve substantially the performance: from 20,000 evaluations NS becomes slower in

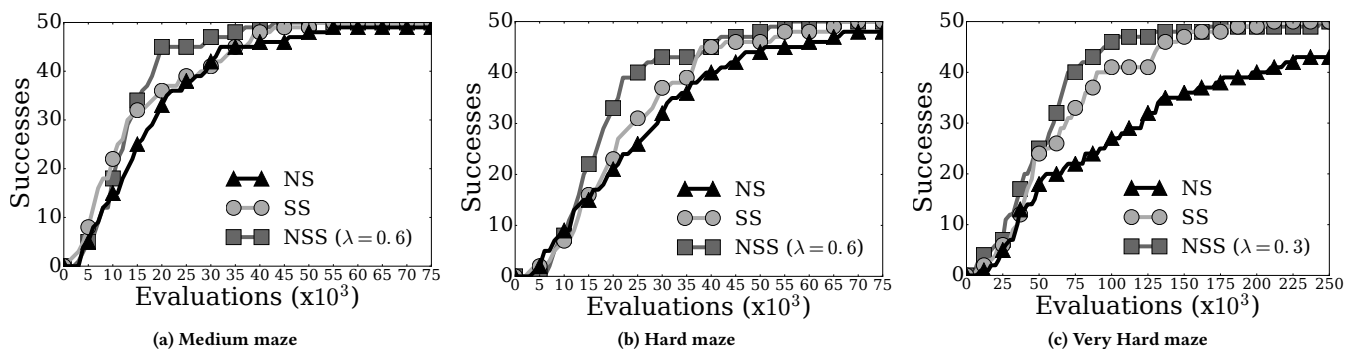


Figure 5: Robustness comparison: number of successes in solving the maze problems over the number of evaluations.

Table 2: Evaluations. Number of evaluations on average to solve the three mazes for each algorithm. Values in parentheses denote the 95% confidence interval.

Maze	Novelty Search	Surprise Search	NSS
Medium	19.0·10 ³ (4.1·10 ³)	15.6·10 ³ (3.5·10 ³)	13.8·10 ³ (2.3·10 ³)
Hard	27.2·10 ³ (5.2·10 ³)	23.7·10 ³ (4.1·10 ³)	19.6·10 ³ (3.3·10 ³)
Very hard	110.4·10 ³ (22.8·10 ³)	68.6·10 ³ (12.7·10 ³)	56.7·10 ³ (11.6·10 ³)

comparison to NSS while SS also fall behind between 10,000 and 35,000 evaluations. As can be seen from Table 2, NSS needs fewer evaluations to solve this maze, and is significantly faster than NS and faster than SS.

As a general conclusion from the robustness comparisons we can clearly observe that coupling novelty and surprise yields better performance both in terms of solutions found and in terms of evaluations for discovering a solution, especially in the two harder mazes where the benefits of NSS are more evident.

5.3 Further analysis



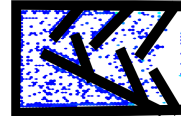





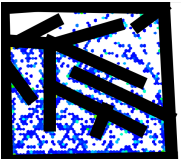

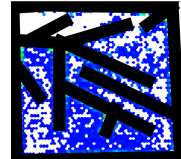

Additional insights on the performance of NSS can be gleaned by analysing the output in the behavioural space as well as the genotypic space. For the former, we observe the heatmaps of robot positions in a number of typical runs for each algorithm in Section 5.3.1. For the latter, we present a number of metrics computed from the final ANNs evolved by the three algorithms in Section 5.3.2.

5.3.1 Behavioural Space: Typical Examples. Table 3 shows a comparison between typical runs for novelty, surprise and NSS, computed from experiments in the three mazes. Each entry describes the number of evaluations (E) taken by the algorithm to find the solution and the heatmap shows the robots' final positions throughout all evaluations. Moreover their corresponding entropy (H) is shown as a measure of their spatial diversity. For each maze, the runs shown are chosen so that the number of evaluations until a solution is discovered are similar among the three algorithms. Not surprisingly, the table shows that NSS is able to explore a larger part of the maze, especially for the hard and very hard mazes: the heatmaps show that NSS results in a more sparse distribution of final robot positions. Furthermore, the entropy values show that, in the hard and very hard maze, the diversity of final positions is

always higher for each typical run considered. Investigating other runs with different E values indicated that the difference in entropy values increases when evolution takes longer to find a solution. Therefore we can infer that combining novelty with surprise is beneficial also for exploring the behavioural space: while novelty searches for unexplored points of the maze, surprise search pushes for unexpected points of the maze, which can involve backtracking to already visited places; their combination augments their respective search capacities.

5.3.2 Genotypic Space. Following the analysis presented in [15], Table 4 shows the metrics collected from the final ANNs evolved by the three algorithms, focusing on two main aspects: their *complexity* and their *diversity*. Genomic complexity is defined as the number of hidden nodes and as the number of connections in the final ANNs, while genomic diversity is measured as the average pairwise distance of the final ANNs evolved. This distance is computed with the compatibility metric, a linear combination of disjoint genes, excess genes and difference in weights, as described in [27]. In terms of genomic complexity, we observe that NSS evolves ANNs larger than novelty search but at the same time smaller than surprise search; the same behaviour can be observed in terms of number of connections. This means that NSS is able to outperform surprise search without creating as complex networks, which alleviates the computational effort needed to evolve complex and large ANNs. In terms of genomic compatibility, NSS does not yield the same diversity as surprise search, but the ANNs are more diverse compared to novelty search. This is probably due to the average size of networks generated by the NSS algorithm. The size affects the disjoint and excess metrics, as it is less probable to have very diverse networks when their size is smaller compared to the ones generated by surprise search.

Table 3: Behavioural Space. Typical successful runs solved after a number of evaluations (E) on the three mazes examined. Heatmaps illustrate the aggregated numbers of final robot positions across all evaluations. Note that white space in the maze indicates that no robot visited that position. The entropy ($H \in [0, 1]$) of visited positions is also reported and is calculated as follows: $H = (1/\log C) \sum_i \{(v_i/V) \log(v_i/V)\}$; where v_i is the number of robot visits in a position i , V is the total number of visits and C is the total number of discretized positions (cells) considered in the maze.

Medium Maze		
Novelty Search	Surprise Search	NSS
		
0  400		
$E = 25000$ $H = 0.63$	$E = 25000$ $H = 0.60$	$E = 25000$ $H = 0.62$
Hard Maze		
Novelty Search	Surprise Search	NSS
		
0  270		
$E = 25000$ $H = 0.61$	$E = 25000$ $H = 0.62$	$E = 25000$ $H = 0.65$
Very Hard Maze		
Novelty Search	Surprise Search	NSS
		
0  330		
$E = 75000$ $H = 0.63$	$E = 75000$ $H = 0.69$	$E = 75000$ $H = 0.71$

6 DISCUSSION

In this paper we have shown that by coupling novelty and surprise search we can obtain an algorithm that is as efficient as and more robust than novelty search or surprise search alone in the maze navigation domain. Combining two different yet powerful ways of divergent search ends up being beneficial with respect to search; the NSS algorithm searches for novel solutions, which means exploring not yet seen points in the search space, and at the same time also rewards surprising solutions, which means deviating from predicted behavioural trends. In fact, when novelty search finds a highly

novel solution, this will likely be still considered novel for several generations until it gains sufficient neighbours. On the other hand, surprise search will consider a solution surprising only in the first generation it appears, as the prediction model will change in the following generations; this has the drawback that a really good solution in terms of global fitness can be eliminated by surprise search because it is not surprising anymore. By aggregating the novelty and surprise scores in NSS, we eliminate both of these drawbacks: novelty is able to “maintain” surprising solutions from evolutionary rejection (if they also have a high novelty score) and, at the same time, very novel solutions have less impact on the selection mechanism as their surprise score degrades throughout evolution. As seen in the behaviour analysis of the testbed navigation task, the coupling is beneficial for search as it pushes robot controllers to spread out and explore the behavioural space more extensively.

While these results already demonstrate the advantages of NSS as a combined form of divergent search, more work needs to be performed to explore the full potential of this coupling. As NSS couples novelty and surprise linearly, it cannot exploit various combinations of novelty and surprise in a dynamic fashion. This static behaviour of NSS limits the capabilities of the algorithm. A direct solution to this limitation is a multiobjective implementation of NSS. Future work will investigate whether using a multiobjective evolutionary algorithm such as NSGA-II [7] can lead to better performance in terms of efficiency and robustness. Moreover future work will explore how NSS can be applied in non-NEAT representations (for example for game weapon generation [14]) and how behaviour characterization can affect the performance of the algorithm.

7 CONCLUSIONS

This paper has introduced the novelty-surprise search algorithm which couples novelty and surprise search and tested its performance in the maze navigation domain. The results show that combining novelty and surprise can be advantageous in terms of efficiency, robustness and behavioural diversity. Furthermore, evolving individuals via NSS exhibits a greater exploratory capacity. Evidently, combining novelty and surprise search allows search to eliminate, in part, limitations inherent in novelty search and surprise search, as it outperforms the performance of the two algorithms when tested on their own.

8 ACKNOWLEDGEMENTS

This work has been supported, in part, by the FP7 Marie Curie CIG project AutoGameDesign (project no: 630665) and the Horizon 2020 project CrossCult (project no: 693150).

REFERENCES

- [1] C. Adami, C. Ofria, and T. C. Collier. Evolution of biological complexity. *Proceedings of the National Academy of Sciences*, 97(9), 2000.
- [2] P. J. Angeline and J. B. Pollack. Competitive environments evolve better solutions for complex tasks. In *Proceedings of the International Conference on Genetic Algorithms*, 1994.
- [3] A. Channon. Passing the ALife test: Activity statistics classify evolution in geb as unbounded. In *Advances in Artificial Life*. Springer, 2001.
- [4] G. Cuccu and F. Gomez. When novelty is not enough. In *European Conference on the Applications of Evolutionary Computation*, pages 234–243. Springer, 2011.
- [5] Y. Davidor. Epistasis variance: A viewpoint on GA-hardness. In *Foundations of Genetic Algorithms*. Morgan Kaufmann, 1991.

Table 4: Genotypic Space. Metrics of genomic complexity and diversity of the final ANNs evolved using NEAT, averaged across successful runs. Values in parentheses denote standard deviations. Significance against novelty ¹, significance against surprise².

Maze	Algorithm	Genomic Complexity		Genomic Diversity			
		Connections	Hidden Nodes	Compatibility	Disjoint	Weight Difference	Excess
Medium	NS	29.0 (6.1)	2.2 (1.0)	32.5 (7.9)	24.9 (7.0)	1.0 (0.2)	4.2 (3.4)
	SS	32.8 (15.0)	2.3 (1.4)	38.8 (19.4)	26.3 (15.3) ¹	1.1 (0.2) ¹	9.07 (10.5) ¹
	NSS	26.4 (6.4) ^{1,2}	2.3 (1.4)	28.7 (7.8) ²	20.9 (7.6) ^{1,2}	1.1 (0.2) ²	4.2 (3.8) ²
Hard	NS	29.7 (8.3)	2.3 (1.1)	34.3 (12.1)	25.5 (10.7)	1.1 (0.2)	5.3 (5.7)
	SS	50.8 (20.7) ¹	3.8 (2.3) ¹	68.7 (25.7) ¹	53.2 (26.9) ¹	1.2 (0.2) ¹	11.7 (12.6) ¹
	NSS	37.3 (15.8) ²	2.8 (1.9) ²	48.0 (20.54) ^{1,2}	36.3 (16.7) ²	1.2 (0.2) ^{1,2}	7.8 (9.2) ^{1,2}
Very Hard	NS	42.0 (14.5)	3.2 (1.9)	56.5 (19.6)	46.6 (19.2)	1.1 (0.2)	6.38 (7.7)
	SS	101.9 (52.4) ¹	6.9 (3.6) ¹	160.3 (67.0) ¹	121.5 (57.1) ¹	1.2 (0.2) ¹	34.9 (37.3) ¹
	NSS	81.1 (46.4) ^{1,2}	5.7 (3.0) ^{1,2}	127.6 (60.2) ^{1,2}	96.1 (50.8) ^{1,2}	1.1 (0.2) ^{1,2}	27.9 (33.2) ^{1,2}

[6] K. Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 3–34. Springer, 2011.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[8] S. G. Ficici and J. B. Pollack. Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states. In *Proceedings of the International Conference on Artificial Life*, 1998.

[9] D. E. Goldberg. Simple genetic algorithms and the minimal deceptive problem. In *Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence*. Morgan Kaufmann, 1987.

[10] D. E. Goldberg and J. H. Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2), 1988.

[11] K. Grace and M. L. Maher. What to expect when you’re expecting: The role of unexpectedness in computationally evaluating creativity. In *Proceedings of the International Conference on Computational Creativity*, 2014.

[12] K. Grace and M. L. Maher. Specific curiosity as a cause and consequence of transformational creativity. *Proceedings of the International Conference on Computational Creativity June*, 2015.

[13] K. Grace, M. L. Maher, D. Fisher, and K. Brady. Modeling expectation for evaluating surprise in design creativity. In *Design Computing and Cognition*. 2014.

[14] D. Gravina, A. Liapis, and G. N. Yannakakis. Constrained surprise search for content generation. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, 2016.

[15] D. Gravina, A. Liapis, and G. N. Yannakakis. Surprise search: Beyond objectives and novelty. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2016.

[16] S. A. Kauffman. *Adaptation on rugged fitness landscapes*. In *Lectures in the Sciences of Complexity*. Addison-Wesley, 1989.

[17] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2), 2011.

[18] J. Lehman and K. O. Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 2011.

[19] J. Lehman and K. O. Stanley. Beyond open-endedness: Quantifying impressiveness. In *Proceedings of the International Conference on Artificial Life*, 2012.

[20] J. Lehman, K. O. Stanley, and R. Miikkulainen. Effective diversity maintenance in deceptive domains. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 2013.

[21] A. Liapis, H. P. Martínez, J. Togelius, and G. N. Yannakakis. Transforming exploratory creativity with DeLeNoX. In *Proceedings of the International Conference on Computational Creativity*, 2013.

[22] A. Liapis, G. N. Yannakakis, and J. Togelius. Constrained novelty search: A study on game content generation. *Evolutionary computation*, 23(1):101–129, 2015.

[23] G. E. Liepins and M. D. Vose. Representational issues in genetic optimization. *Journal of Experimental and Theoretical Artificial Intelligence*, 2(101), 1990.

[24] J.-B. Mouret and S. Doncieux. Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evolutionary computation*, 20(1):91–133, 2012.

[25] J. K. Pugh, L. B. Soros, and K. O. Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

[26] C. R. Reeves. *Fitness landscapes*. In *Search Methodologies*. Springer, 2005.

[27] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2), 2002.

[28] R. Velez and J. Clune. Novelty search creates robots with general skills for exploration. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 737–744. ACM, 2014.

[29] T. Weise, R. Chiong, and K. Tang. Evolutionary optimization: Pitfalls and booby traps. *Journal of Computer Science and Technology*, 27(5):907–936, 2012.

[30] S. Wessing, M. Preuss, and G. Rudolph. Niching by multiobjectivization with neighbor information: Trade-offs and benefits. In *Proceedings of the Evolutionary Computation Congress*, 2013.

[31] L. Yaeger. Poly world: Life in a new context. *Proceedings of Artificial Life*, 3, 1994.

[32] G. N. Yannakakis and A. Liapis. Searching for surprise. In *Proceedings of the International Conference on Computational Creativity*, 2016.