Erschienen in: Proceedings of the ALIFE 2020 : The 2020 Conference on Artificial Life / Bongard, Josh; Lovato, Juniper; Soros, Lisa et al. (Hrsg.). - Cambridge, Massachusetts : MIT Press, 2020. - S. 384-392
https://dx.doi.org/10.1162/isal_a_00266

Evolution of Diverse Swarm Behaviors with Minimal Surprise

Tanja Katharina Kaiser and Heiko Hamann

Institute of Computer Engineering, University of Lübeck, 23562 Lübeck, Germany
{kaiser, hamann}@iti.uni-luebeck.de

Abstract

Complementary to machine learning, controllers for swarm robotics can also be evolved using methods of evolutionary computation. Approaches such as novelty search and MAP-Elites go beyond mere fitness-based optimization by increasing the time spent on exploration. Instead of optimizing a fitness function, selective pressure towards unexplored behavior space is generated by forcing behavioral distance to previously seen behaviors. Ideally, we would like to define a generic behavioral distance function; however, effective distance functions are usually domain specific.

Our minimize surprise approach concurrently evolves two artificial neural networks: one for action selection and one as world model. Selective pressure is implemented by rewarding good predictions of the world model. As an effect, the evolutionary dynamics push towards swarm behaviors that are easy to predict, that is, the robots virtually try to minimize surprise in their environment. Here, we compare minimize surprise to novelty search and, as baseline, a genetic algorithm in simulations of swarm robots. We observe a diversity of collective behaviors, such as aggregation, dispersion, clustering, line formation, etc. We find that minimize surprise is competitive to novelty search for the investigated swarm scenario, although it does not require a cleverly crafted domain-specific behavioral distance function.

Introduction

In classical evolutionary swarm robotics (Trianni, 2008), standard methods of evolutionary computation are applied to generate swarm robot controllers similarly to evolutionary robotics (ER), an approach that has considerable successes (Duarte et al., 2016a,b). It enables the evolution of controllers for several swarm robotics tasks such as aggregation (Trianni et al., 2003), foraging (Liu et al., 2007) or group transport (Groß and Dorigo, 2008). A variety of different approaches have been proposed, such as combining learning and evolution (Nolfi et al., 1994). However, ER turned out not to scale well with task complexity and other methods were developed that moved the scope towards increased diversity and exploration. Most prominently, this is implemented by novelty search (Lehman and Stanley, 2008) and later by more sophisticated methods, such as MAP-Elites (Mouret and Clune, 2015). Novelty search is also applied in swarm robotics (Gomes et al., 2015). Approaches that balance search for quality and for diversity are called quality diversity algorithms (Pugh et al., 2016). Despite their successes, a disadvantage is that, for example in novelty search, the fitness function is substituted by a behavioral distance function that is again task-specific. Mouret and Doncieux (2012) discuss the analogy between fitness function design (Nelson et al., 2009) and behavioral distance measure design: "novelty search users have to craft the behavior distance to create a [. . .] gradient [. . .] In an ideal ER setup, ER researchers would only define a high-level fitness function [. . .] This goal could be achieved with a generic behavioral distance function." However, it seems unlikely that such a generic behavioral distance function exists.

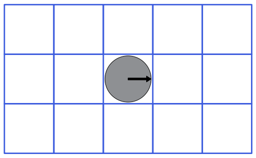


Figure 1: Sensor model

Our approach of minimize surprise (Hamann, 2014) may have more potential to be generic. In addition to evolving an artificial neural network (ANN) for action selection, also a world model is evolved as ANN. This world model is rather simple and needs to predict the robot's sensor input for the next time step given its current sensor input and chosen action. The evolution is then driven by the intrinsic motivation of rewarding correct predictions of the world model while both ANN are mutated. As a consequence the action selection network receives only indirect selective pressure (e.g., a mutation changes the robot's behavior to better fit its predictions). While this way the approach is task-agnostic and generic, the evolutionary dynamics are biased towards 'boring behaviors,' that is, easy to predict collective behaviors. As it turns out, these are then natural swarm behaviors (Hamann, 2014) and useful self-assembly behaviors (Kaiser and Hamann, 2019). The application of the minimize surprise approach is hence similar to novelty search. After the evolutionary runs, users need to inspect the obtained variety of behaviors and pick those that suit their needs. Alternatively, this process can be automatized in the form of an automatic post-evaluation using a task-specific fitness function. Another relevant approach is that of Turing learning (Groß et al., 2017) where in a conceptually similar technique, robots need to adapt their behaviors to a swarm while being observed by an adversarial agent who tries to identify them.

Here, we are interested in collective robot behavior (Hamann, 2018) and we compare different evolutionary methods in a simple grid-world simulator. Our main research question is to find whether novelty search or minimize surprise, plus a genetic algorithm (GA) as baseline, generate more diverse behaviors of high quality in the given swarm scenario. This study has two characteristic challenges. (A) The choice of a behavioral distance measure (and a fitness function for the GA) is potentially arbitrary. (B) We need to quantify the algorithms' creative results. We address challenge A by testing two behavioral distance measures for novelty search. We address challenge B by using classification scripts that are inspired by experience about the complete set of behaviors emerging in this scenario (Kaiser and Hamann, 2019).

Our main results are that minimize surprise is competitive to novelty search in diversity but can provide more quality while novelty search is sensitive to its behavioral distance function and more biased towards exploration. The challenge for applying minimize surprise is to properly parameterize the environment, for example, to provoke the emergence of interesting behaviors by a critical swarm density. The challenge for novelty search is to find a generic behavioral distance function or at least a high performing function for a given scenario.

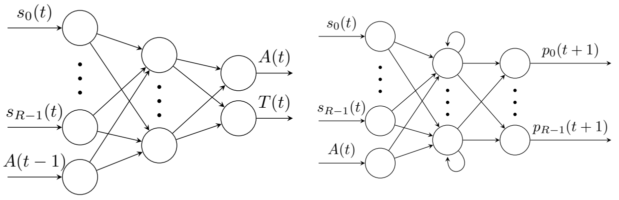
Methods

We compare five different evolutionary approaches to evolve collective robot behaviors in a simple grid-world simulation: (1) a fitness-based GA as a baseline, (2) minimize surprise (MS), (3) minimize surprise with predefined predictions (MS-P), (4) novelty search using domain-dependent behavioral characteristics (N-CoM), and (5) novelty search using domain-independent behavioral characteristics (N-SV). Thus, we extend our previous work by (3) and (5) (Kaiser and Hamann, 2020).

Simulation Environment

In all experiments, our swarm of robots lives on a square 2D grid with toroidal boundary conditions (torus). We vary swarm density (i.e., robots per area: $\frac{N}{G \times G}$) by keeping the swarm size $N = 100$ fixed and changing the grid side lengths G as in previous work (Kaiser and Hamann, 2019). This leads to swarm densities of 44% ($\frac{100}{15 \times 15}$) and 25% ($\frac{100}{20 \times 20}$), respectively.

Each robot has $R = 14$ binary sensors to detect robots in its surrounding grid cells, see Fig 1. In each time step, a robot can either move one grid cell forward or rotate by $\pm 90°$. So the robots have discrete headings: North, South, East and West. We restrict the number of robots per grid cell to one. An intended step forward is blocked if the targeted grid cell is occupied. In that case, the robot stays stopped on its current grid cell. A robot's action is determined by a feed-forward ANN controller (see Fig. 2a) with 15 inputs (14 sensor values plus the action value of the last time step), eight hidden neurons and two outputs. The outputs are an action value determining whether the robot moves forward or turns and the turning direction. Genomes encode the ANN weights as floating point values and are initialized in the range $[-0.5, 0.5]$. We use a homogeneous swarm in two regards: first, all robots of an evaluation use the same genome and second, all robots have the same capabilities.



(a) controller

(b) world model (MS)

Figure 2: ANNs used in the evolutionary algorithms. Inputs are R sensor values $s_0(t), \ldots, s_{R-1}(t)$ at time step t, action value $A(t-1)$ of time step $t-1$ or $A(t)$ of time step t, respectively. Outputs are action value $A(t)$ and turning direction $T(t)$ or R sensor value predictions $p_0(t+1), \ldots, p_{R-1}(t+1)$ for time step $t+1$. World model only used in minimize surprise.

Table 1: Parameters

Parameter	Value
elitism	1
mutation rate	0.1
population size	50
generations	100
evaluation length (time steps)	500
repetitions	10

Evolutionary Algorithms

All five approaches are based on a standard genetic algorithm. We use the same parameters in all experiments as summarized in Tab. 1. We use proportionate selection and do not use crossover. Mutation adds a random number between $[-0.4, 0.4]$ to the respective gene. Elitism is not applied to novelty search. Each approach is evaluated in 50 independent evolutionary runs. We position robots randomly in the beginning of each evaluation.

Genetic Algorithm A standard GA with a task-specific fitness function is used as a baseline for comparison. In general, we do not aim for a specific behavior, but aim for behavioral diversity. As we have to use a task-specific fitness function in GA, we choose arbitrarily to evolve aggregation behaviors. The fitness function rewards shorter average distances to the center of (robot) mass in the last time step (a common choice (Trianni et al., 2003; Gomes et al., 2013)). Fitness F is defined as

$$F = \frac{1}{N} \sum_{n=0}^{N-1} 1 - \text{dist}(CoM(T), pos_n(T)) \ , \qquad (1)$$

with swarm size N, center of mass $CoM(T)$ and position $pos_n(T)$ of robot n in the last time step T of the evaluation. We calculate the center of mass on the 2D grid with toroidal boundary conditions based on Bai and Breen (2008). Due to discrete headings and movement directions of robots, we use the Manhattan distance to calculate distances between center of mass and robots. Distance values are normalized to $[0, 1]$. The minimum fitness value obtained in ten repetitions per genome is assigned as fitness to the individual.

Minimize Surprise Our approach of minimal surprise (Hamann, 2014; Kaiser and Hamann, 2019) allows the evolution of robot (swarm) behaviors solely using an intrinsic driver, here to reach high sensor value prediction accuracy. Each robot in the swarm is equipped with two ANN (see Fig. 2) and shares the genome with all other swarm members. Additional to the feed-forward ANN serving as the robot controller (Fig. 2a), robots have a recurrent ANN as a world model (Fig. 2b) that predicts the sensor values of the next time step. It has 15 inputs (14 sensor values plus next action $A(t)$), 15 hidden neurons and 14 outputs for the sensor value predictions.

The fitness function rewards prediction accuracy and is defined over the evaluation period of T time steps as

$$F = \frac{1}{TNR} \sum_{t=0}^{T-1} \sum_{n=0}^{N-1} \sum_{r=0}^{R-1} 1 - |p_n^r(t) - s_n^r(t)| \ , \qquad (2)$$

with swarm size N, number of sensors per robot R, prediction $p_n^r(t)$, and sensor value $s_n^r(t)$ for sensor r of robot n at time step t. We have selective pressure on the world model while the controller receives no direct rewards. As the networks are evolved in pairs, combinations of controller and world model gaining high fitness are more likely to survive in the evolutionary process.

As a variant of the approach, we can engineer self-organization by (partially) predefining sensor values (i.e., fixing all or several outputs of the world model). Assuming a certain behavior is associated with a static sensor input pattern, we can predefine what we want the robot to perceive in a swarm showing the desired collective behavior. Robots would then receive rewards once they evolve a behavior that generates that sensor input pattern. Thus we push evolution towards the emergence of desired behaviors without having to tailor a task-specific fitness function (Nelson et al., 2009). This allows a gradient from running the approach with complete freedom (no sensor prediction predefined) to simplifying it to a special kind of fitness-based GA (all sensor predictions predefined). In the latter case, all outputs of the world model are fixed making the respective ANN obsolete. Fitness is then directly defined via the desired sensor pattern. These predefined predictions serve as a local template for the behavior, for example, defining a targeted structure in self-assembly that is similar to approaches used in cellular automata (Hoffmann and Désérable, 2016). We exemplified minimize surprise with predefined predictions aiming for the emergence of line structures in a robot swarm self-assembly approach (Kaiser and Hamann, 2019).

We run two experiments with minimize surprise. (1) The standard minimal surprise approach (MS) and (2) predefining all sensor predictions to 1 (cf. Fig. 1, MS-P) to aim for grouping behaviors as with the fitness-based GA. In both experiments, the minimum fitness value reached in ten repetitions per genome is assigned to an individual as fitness.

Novelty Search An approach that continually produces novel individuals instead of driving the evolutionary process towards a fixed goal is novelty search (Lehman and Stanley, 2008). Behavioral (i.e., phenotypic) diversity is promoted by scoring individuals on how different their behavior is compared to previously found individuals.

The novelty ρ of an individual is calculated by a novelty metric that determines the sparseness at a point in behavior space taking into account the current population and an archive of past individuals. These samples of individuals represent previously visited regions of search space and the current list of solution candidates. Only a subset of explored individuals is added to the archive to limit the algorithm's computational complexity. There are several strategies with different conditions when to add behaviors to the archive. (A) The calculated novelty ρ is above a minimum threshold $\rho > \rho_{\text{min}}$; (B) each individual can be added with low probability, or (C) a combination of both. Lehman and Stanley (2010) promote approach B because always adding individuals of high novelty may limit the local search in newly explored regions of search space.

Novelty (or sparseness) of an individual x in behavior space is given by

$$\rho(x) = \frac{1}{K} \sum_{k=0}^{K-1} \text{b\_dist}(x, \mu_k) \ , \qquad (3)$$

where μ_k is the kth-nearest neighbor of individual x with respect to the behavioral distance metric $\text{b\_dist}(\cdot, \cdot)$ with a maximum of K nearest neighbors. The behavior of an individual is characterized by a vector of domain-dependent behavioral features that are supposed to capture relevant aspects of an assumed task. Key is to carefully select these features such that useful and diverse behaviors are found and distinguished. While the full description of an observed behavior is complex and high-dimensional, the dimension reduction by selected features may project classes of behaviors onto the same feature vector. An unfortunate selection of features may hence be blind to certain behavioral differences and consequently be incapable to discover all relevant behaviors. Once the features are defined, the behavioral distance is commonly defined by the Euclidean distance between two feature vectors.

Lehman and Stanley (2008) showcased novelty search for single agents, for example in a maze navigation task. The most relevant work here out of the vast literature on novelty search is that by Gomes et al. (2013) who applied it successfully to swarm robotics by evolving controllers for aggregation and resource sharing.

We run two novelty search experiments with different behavioral characteristics. First, we use distance to center of mass averaged over all robots in the last time step of an evaluation as a domain-specific behavioral feature as in the fitness-based GA. We define a 1D behavioral vector

$$b_{CoM} = \left[\frac{1}{N}\sum_{n=0}^{N-1} \text{dist}(CoM(T), pos_n(T))\right] . \tag{4}$$

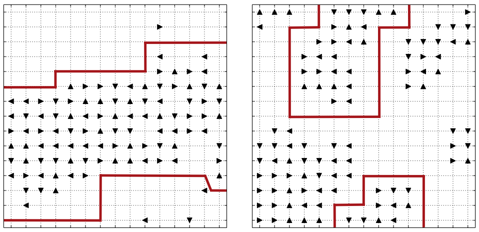
In a second variant of the novelty search experiment, we use a vector of sensor values averaged over all robots in the last time step as a potentially task-independent behavioral characteristic defined as an R-dimensional vector

$$b_s = \frac{1}{N}\left[\sum_{n=0}^{N-1} s_n^0(T), \ldots, \sum_{n=0}^{N-1} s_n^{R-1}(T)\right] . \tag{5}$$

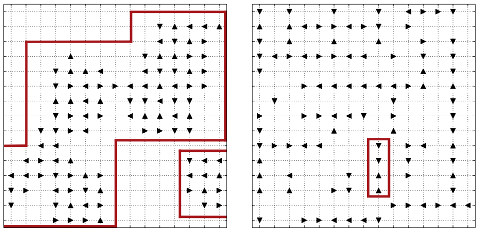
Similarly to our concept in minimize surprise, sensor values can serve as local templates of forming patterns and thus, might enable the evolution of a variety of structures. Novelty search tries to popularize all variants of these vectors which should correspond to generating different patterns formed by robots. For both novelty search variants, the behavioral vectors obtained in ten repetitions are averaged. We add behaviors to the archive with a probability of 2% and use up to the tenth nearest neighbor ($K = 10$ in Eq. 3).

Evaluation of Resulting Structures

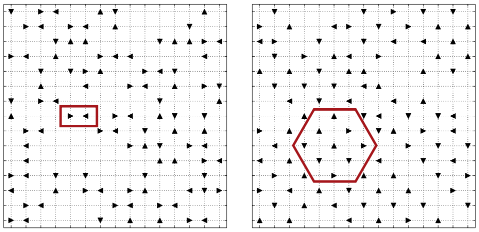
To compare the variety of emergent patterns, we output one robot trajectory per best evolved individual (minimize surprise, GA) or per evaluated individual (novelty search). All evolved behaviors are then post-evaluated considering the robot positions, that is, the formed structures, in the last time step of an evaluation using Python scripts[1] as in previous work (Kaiser and Hamann, 2019). Structures are classified based on their highest resemblance to one of eight classes: aggregation, clustering, loose grouping, lines, pairs, triangular lattice, squares, and random dispersion. This is a conservative classification because low competence of forming a certain recognized pattern is acknowledged. Even if only few robots of a swarm form recognizable structures, then the respective genome is classified as that behavior. Hence, we use the percentage of a swarm's robots forming a considered structure to assess the respective behavior's quality. Example structures for each class are shown in Fig. 3.



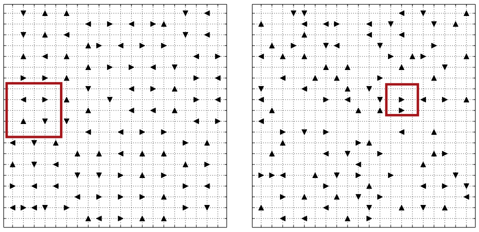
(a) aggregation (b) cluster



(c) loose grouping (d) lines



(e) pairs (f) triangular lattice



(g) squares (h) random dispersion

Figure 3: Examples of formed structures. Triangles represent robots and their headings. Red boxes indicate one arbitrary sample of the relevant pattern to guide the eye.

There are three grouping behaviors: aggregation (Fig. 3a), clustering (Fig. 3b), and loose grouping (Fig. 3c). A robot

---

[1] `https://github.com/msminirobot/minimal-surprise-self-assembly`

with at least three neighbors in its von Neumann neighborhood and at least six neighbors in its Moore neighborhood is considered to be part of a cluster. All of the robot's neighbors are then considered part of the cluster, too. Clustering is the formation of several separate clusters while one single cluster is formed in aggregation. Several completely interconnected clusters are classified as loose grouping.

Pairs (Fig. 3e) and lines (Fig. 3d) are structures that consist of two or more robots, respectively. Robots are required to have parallel headings and the structure has to be terminated by robots pointing inwards (i.e., a static robot formation as robot movements are blocked). Furthermore, the number of allowed neighbors on each side next to the structure is limited to half of the structure's length.

We distinguish three dispersion behaviors: triangular lattice (Fig. 3f), squares (Fig. 3g), and random dispersion (Fig. 3h). To be classified as triangular lattice, robots have to be positioned in a 2D hexagonal lattice of points. Squares have robots on every second grid cell, that is, robots are one grid cell apart. We loosen the criteria for random dispersion compared to previous work to adapt to higher swarm densities. We classify a structure as randomly dispersed if robots have no neighbors in their von Neumann neighborhood or at maximum one neighbor in their Moore neighborhood.

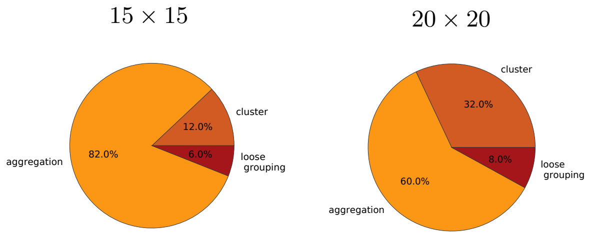
Results

We compare all approaches based on the generated behavioral diversity extending our previous work (Kaiser and Hamann, 2020) by novelty search with b_s (N-SV) and minimize surprise with predefined predictions (MS-P). In addition, we check the influence of swarm density on minimize surprise and compare the runtime of the different approaches.

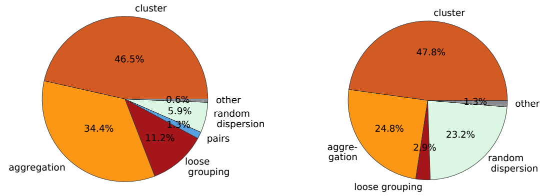
Resulting structures We classify the resulting behaviors and evaluate the solution quality measuring the percentage of robots in the formed structures at the evaluation's last time step T of the 50 best evolved individuals in fitness-based GA and the two minimize surprise variants. In novelty search, all individuals are potential solutions. Thus, we post-evaluate all 250,000 individuals per novelty search approach (50 individuals, 100 generations, 50 experiments). The percentages of found structures for each of the five approaches are shown as pie charts in Fig. 4. Fig. 5 gives boxplots of the solution quality per approach and grid size.

We compare the behavior distributions of all five approaches (Fig. 4) pairwise for both grid sizes. We find statistically significant differences for all pairs (Fisher's Exact Test with $p < 0.01$) except for fitness-based GA and minimize surprise with predefined predictions (MS-P) on the 15×15 grid.

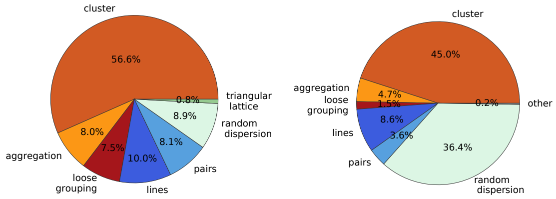
Both in fitness-based GA and minimize surprise with predefined predictions, we arbitrarily aim for the emergence of grouping behaviors (clustering, aggregation, loose group-



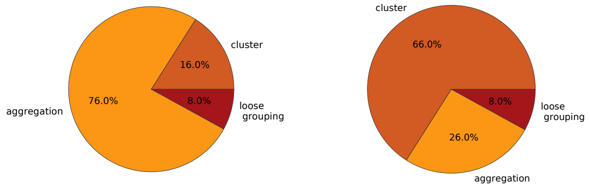
(a) GA (b) GA



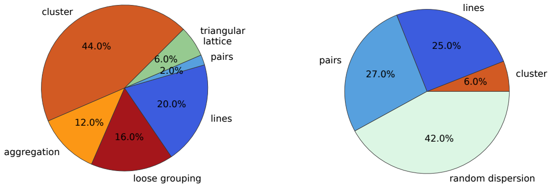
(c) N-CoM (d) N-CoM



(e) N-SV (f) N-SV

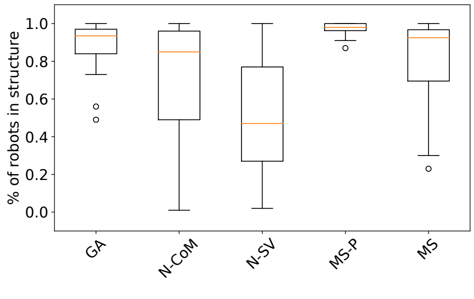


(g) MS-P (h) MS-P

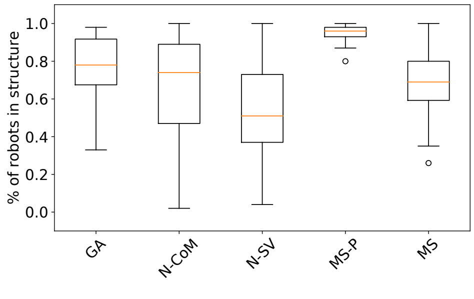


(i) MS (j) MS

Figure 4: Resulting structures for fitness-based GA, novelty search with task-specific behavioral characteristic b_{CoM} based on robot distances to center of mass (N-CoM), novelty search with potentially task-independent behavioral characteristic b_s based on sensor values averaged over all robots (N-SV), minimize surprise with predefined predictions (MS-P), and minimize surprise (MS) per grid size. *Other* represents structures that occurred in less than 1% – in (c): 0.25% triangular lattices and 0.4% lines; in (d) 0.07% triangular lattices and 0.02% squares; in (f) 0.15% triangular lattices and 0.09% squares. Figs. a-d, i&j reprinted from Kaiser and Hamann (2020).



(a) 15 × 15 grid



(b) 20 × 20 grid

Figure 5: Solution quality measured as percentage of robots positioned in the structure for fitness-based GA, novelty search with b_{CoM} (N-CoM), novelty search with b_s (N-SV), minimize surprise with predefined predictions (MS-P), and minimize surprise (MS). All pairs are statistically significant different except for MS&GA on the 15 × 15 grid, and MS&N-CoM on the 20 × 20 grid (Mann-Whitney U Test, $p < 0.05$).

ing). As expected, both approaches produce those behaviors due to the task-specific fitness functions. While the results on the smaller grid are similar (Figs. 4a and 4g, Jensen-Shannon divergence (JSD; i.e., similarity between two probability distributions) between GA and MS-P of approx. 0.004), we notice a statistically significant difference in the structure distributions on the larger grid (Figs. 4b and 4h). Fitness-based GA leads to a majority of aggregation behaviors, but clustering prevails in minimize surprise with predefined predictions. The different nature of the fitness functions seems to cause that difference. In fitness-based GA, we have a global view on aggregation by calculating the distance to the center of mass (Eq. 1). In contrast, using the fitness function of minimize surprise (Eq. 2) with predefined sensor predictions equates to using a local template for behaviors, here, grouping behaviors. Even distant clusters can have high fitness scores. On the larger grid (i.e., the lower swarm density setting), clustering emerges more easily. Swarm density and fitness function combined cause different behavior distributions. The solution quality of emergent grouping behaviors is significantly higher for minimize surprise with predefined predictions (MS-P) compared to GA on both grids (Mann-Whitney U Test with $p < 0.01$). The median percentage of robots in the structure is higher for MS-P than for GA (Fig. 5). Interquartile range and minimum are smaller. Minimize surprise with predefined predictions offers a more intuitive way to target behaviors and allows for the evolution of high-quality solutions.

Comparing both novelty search variants with each other and with minimize surprise leads to a statistically significant difference in the distribution of resulting structures (Fisher's Exact Test with $p < 0.01$). Here, we measure the Jensen-Shannon divergence to a maximally diverse behavior distribution, that is, a uniform distribution of 12.5% for each of all eight structures. The behavior distribution in minimize surprise (JSD ≈ 0.24) is closer to a maximally diverse distribution than that of novelty search with b_{CoM} on 15 × 15 (Eq. 4, JSD $\approx$ 0.33). The distributions of minimize surprise and novelty search with b_s are similarly close to maximum diversity (JSD ≈ 0.23). Minimize surprise and novelty search with b_{CoM} lead to similarly diverse behavior distributions on 20 × 20 (JSD ≈ 0.35 both) while novelty search with b_s is more diverse (JSD ≈ 0.3). Novelty search leads to a greater variety of structures than minimize surprise. This comparison is arguably biased because for novelty search we include 5000 times more genomes (250,000 individuals as potential solutions in novelty search versus the 50 best individuals in minimize surprise). We would expect additional behavior types generated by chance. As above, we notice an influence of the swarm density on the resulting behaviors. In higher swarm densities, grouping behaviors are easier to form while for lower swarm densities we see dispersion behaviors. But for novelty search with b_{CoM} on both grids, we find mainly grouping behaviors caused by domain specific behavior characterization (distance to center of mass).

Considering the solution quality (Fig. 5) of the 50 best individuals (GA, MS, MS-P) or all individuals (N-CoM, N-SV), minimize surprise reaches significantly higher percentages of robots in the structure than novelty search with b_s on both grid sizes (Mann-Whitney U Test with $p < 0.01$). The solution quality is significantly greater than for novelty search with b_{CoM} on the smaller grid (Mann-Whitney U Test with $p < 0.01$), while we do not find a statistically significant difference on 20 × 20. The boxplots in Fig. 5 show that minimize surprise (MS) has the maximum median on the smaller grid, while it reaches a similar level as novelty search with b_{CoM} (N-CoM) and a higher median than novelty search with b_s (N-SV) on 20 × 20. On both grids the interquartile range is smaller and the minima are greater for minimize surprise (MS) than for both novelty search variants. All three approaches reach similar maximum percentages of robots in the structure. Novelty search with b_s leads to greatest variety of emerging structures, but to the worst solution quality indicating the tradeoff between exploration and exploitation.

Swarm density & emergent structures in minimize surprise We find a statistically significant difference for minimize surprise on the 15×15 and 20×20 grids (Fisher's Exact Test, $p < 0.01$). While on the smaller grid mainly grouping behaviors emerge, a tendency towards dispersion can be found on the larger grid. We run 50 independent experiments for each square grid size between 10×10 and 30×30[2]. Thus, we examine swarm densities between 100% ($\frac{100}{10\times10}$) and $\approx 11\%$ ($\frac{100}{30\times30}$). The resulting behavior distributions are visualized as bar charts in Fig. 6.

With increasing grid size, we find the behavior distribution shifting from mainly grouping behaviors to dispersion as well as pairs and lines. This is intuitive as grouping behaviors easily form in high swarm densities and are easy to predict. With lower swarm density, we see more random dispersion, squares, lines, and pairs. In general, the Jensen-Shannon divergence between two swarm density settings increases with the difference in swarm density, that is, the distributions are less similar. The highest and lowest swarm density settings reach a maximum JSD of 1.0 and are maximally different. This is intuitive as these distributions consist of solely one to two behaviors. For grids $G \in \{10, 11, 12\}$ only aggregation and loose grouping emerge as movement is barely possible. Random dispersion and pairs form on grids $G \in \{29, 30\}$. We find that the distribution evolved on 16×16 seems rather invariant to swarm density with minimum JSD 0.06 (15×15) and maximum 0.83 (30×30). With seven different behaviors, this setting also has the most diverse behavior distribution.

Runtime Comparison We compare runtimes of all five approaches because minimize surprise requires calculations for two ANN. We limit file I/O to a bare minimum. We run each approach five times on a MacBook Pro (2017) with a 3.1 GHz Intel Core i5 processor (7th generation) and 16 GB RAM for the 15×15 grid case. The fitness-based GA and minimize surprise with predefined predictions have roughly the same runtime ($\approx$ 23 min). The two novelty search runs require slightly more runtime ($\approx$ 26.5 min), probably caused by the increased file I/O. The standard minimize surprise approach needs about twice the runtime of the other approaches (≈ 55 min). We need to calculate outputs of a second ANN and thus, the runtime basically doubles.

We have to take into account not only the runtime, but also the post-evaluation effort. Novelty search requires the post-evaluation of all individuals to find the best evolved controller for a given task. Based on the parameters of the evolutionary run, that is, generations and population size, this post-evaluation can be a tedious task that would be computationally expensive when automated.

Conclusion

We have answered our main research question by our finding that minimize surprise is competitive to novelty search concerning the created diversity of behaviors while providing behaviors of good quality. We have tested two behavioral distance functions in the case of novelty search. One can argue that a non-tested third function would have resulted in different findings. However, even if a behavioral distance function exists that would have let to better results for novelty search, finding an optimal function would be a similar challenge in an actual application. The problem of task-specific behavioral distance functions is hence a problem in terms of both the methodology of our study here and the implementation in a use case.

The classification of behaviors may seem arbitrary. At least where to draw the boundaries between similar behaviors, such as loose grouping and clustering, is possibly subjective. Also we cannot guarantee that we have identified all observable behaviors that can be distinguished from others. Still, the behavior classification approach is intuitive and our results are most likely robust to reasonable changes in the classification scheme (e.g., removing one class and adding another). Another important factor for classification are the initial robot positions. We showed in previous work (Kaiser and Hamann, 2019) that evolved behaviors for line formation are robust against different initial positions. However, some patterns, such as aggregation, clustering, and loose grouping, are similar to each other and their formation might be influenced by initial robot positions. To guarantee the formation of one of these structures, we would require a post-evaluation with different initial starting positions that can be time-consuming for large sets of potential solutions.

A drawback of minimize surprise compared to novelty search is that it has no explicit driver for diversity. Diversity in minimize surprise is a response to the environment (e.g., swarm density). If a trivial behavior allows for immediate prediction success in a given environment, then most evolutionary runs deliver this uninteresting behavior. Hence, the generic feature of rewarding correct predictions is partially counteracted by finding appropriate setups and environments. Whether this is a true challenge for applications or whether real applications have enough complexity to trigger diversity will be part of our future work.

A qualitative difference between minimize surprise and novelty search is that novelty search creates behavioral diversity within an evolutionary run while minimize surprise does that across independent runs. That is intriguing because minimize surprise creates diverse evolutionary dynamics of complete populations fully depending on the random initialization and guided in a self-organized process. Novelty search instead diversifies populations by an explicitly implemented selective pressure away from seen solutions. One can argue that minimize surprise natively implements a quality diversity approach (quality within runs and diversity

---

[2] We do not run the 10×10 setting with swarm density 100% (100 robots on 100 grid cells) as that would trivially allow only for aggregation.

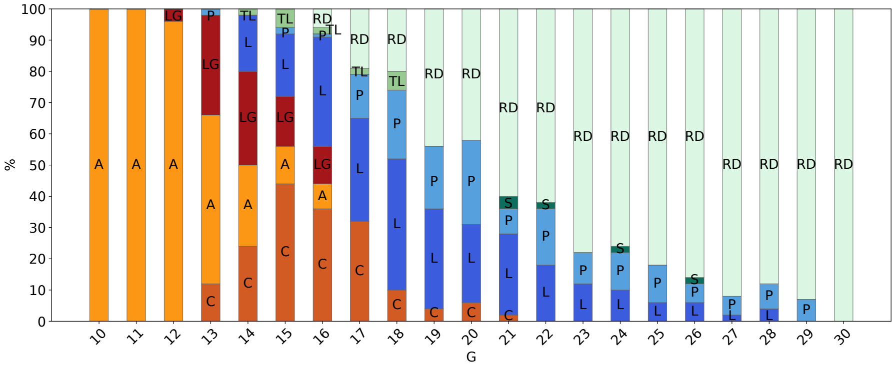


Figure 6: Percentage of resulting structures in minimize surprise for different grid sizes $G \times G$ with clustering (C), aggregation (A), loose grouping (LG), lines (L), pairs (P), triangular lattice (TL), squares (S), and random dispersion (RD).

across runs) while novelty search needs to be augmented by an explicit local search (Pugh et al., 2016). However, novelty search can be seen as the more user-friendly approach as it provides a diverse set of potential solutions after a single run. This set of potential solutions has to be post-evaluated to find a behavior suitable for a given task and providing an acceptable solution quality. We identify a tradeoff between running only a single run and putting much time into selecting an appropriate behavior as in novelty search or running several runs and picking the best option out of a few high quality behaviors in minimize surprise.

We have shown that predefining sensor predictions in minimize surprise provides an intuitive and easy-to-use way to define a task-specific fitness function at least for simple tasks. Evolution is pushed towards the emergence of specific behaviors and even leads to higher quality solutions than GA with a hand-crafted fitness function for the desired behavior.

We have employed a simple sample scenario in a grid-world simulation to compare five approaches. We aim to bridge the reality gap in future work via a robotics simulator to real robot experiments. We have studied patterns formed by robots that result in time-invariant sensor input. In a future scenario, we plan to introduce additional complexity that may result in interesting behaviors with time-variant sensor inputs. Our final objective is to test and compare novelty search and minimize surprise on real robots.

References

Bai, L. and Breen, D. (2008). Calculating center of mass in an unbounded 2d environment. *Journal of Graphics Tools*, 13(4):53–60.

Duarte, M., Costa, V., Gomes, J., Rodrigues, T., Silva, F., Oliveira, S. M., and Christensen, A. L. (2016a). Evolution of collective behaviors for a real swarm of aquatic surface robots. *PLOS ONE*, 11(3):1–25.

Duarte, M., Costa, V., Gomes, J., Rodrigues, T., Silva, F., Oliveira, S. M., and Christensen, A. L. (2016b). Unleashing the potential of evolutionary swarm robotics in the real world. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, GECCO'16 Companion, pages 159–160, New York, NY, USA. ACM.

Gomes, J., Mariano, P., and Christensen, A. L. (2015). Cooperative coevolution of partially heterogeneous multiagent systems. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 297–305. International Foundation for Autonomous Agents and Multiagent Systems.

Gomes, J., Urbano, P., and Christensen, A. L. (2013). Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7(2-3):115–144.

Groß, R. and Dorigo, M. (2008). Evolution of solitary and group transport behaviors for autonomous robots capable of self-assembling. *Adaptive Behavior*, 16(5):285–305.

Groß, R., Gu, Y., Li, W., and Gauci, M. (2017). Generalizing GANs: A Turing perspective. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6319–6329.

Hamann, H. (2014). Evolution of collective behaviors by minimizing surprise. In Sayama, H., Rieffel, J., Risi, S., Doursat, R., and Lipson, H., editors, *14th Int. Conf. on the Synthesis and Simulation of Living Systems (ALIFE 2014)*, pages 344–351. MIT Press.

Hamann, H. (2018). *Swarm Robotics: A Formal Approach*. Springer.

Hoffmann, R. and Désérable, D. (2016). Line patterns formed by cellular automata agents. In El Yacoubi, S., Was, J., and Bandini, S., editors, *Cellular Automata*, pages 424–434, Cham. Springer International Publishing.

Kaiser, T. K. and Hamann, H. (2019). Engineered self-organization for resilient robot self-assembly with minimal surprise. *Robotics and Autonomous Systems*, 122:103293.

Kaiser, T. K. and Hamann, H. (2020). Diversity in swarm robotics with task-independent behavior characterization. In *Genetic and Evolutionary Computation Conference Companion*, GECCO '20 Companion, New York, NY, USA. ACM. [extended abstract]. `https://doi.org/10.1145/3377929.3389949`.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In Bullock, S., Noble, J., Watson, R., and Bedau, M. A., editors, *Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, pages 329–336. MIT Press.

Lehman, J. and Stanley, K. O. (2010). Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, GECCO '10, page 837–844, New York, NY, USA. Association for Computing Machinery.

Liu, W., Winfield, A., and Sa, J. (2007). Modelling swarm robotic systems: A case study in collective foraging. *Towards Autonomous Robotic Systems (TAROS 2007)*.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

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

Nelson, A. L., Barlow, G. J., and Doitsidis, L. (2009). Fitness functions in evolutionary robotics: A survey and analysis. *Robotics and Autonomous Systems*, 57:345–370.

Nolfi, S., Parisi, D., and Elman, J. L. (1994). Learning and Evolution in Neural Networks. *Adaptive Behavior*, 3(1):5–28.

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

Trianni, V. (2008). *Evolutionary Swarm Robotics - Evolving Self-Organising Behaviours in Groups of Autonomous Robots*, volume 108 of *Studies in Computational Intelligence*. Springer, Berlin, Germany.

Trianni, V., Groß, R., Labella, T. H., Şahin, E., and Dorigo, M. (2003). Evolving aggregation behaviors in a swarm of robots. In Banzhaf, W., Ziegler, J., Christaller, T., Dittrich, P., and Kim, J. T., editors, *Advances in Artificial Life (ECAL 2003)*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 865–874. Springer.