# Swarm Intelligence Approach in Detecting Spatially-Independent Symmetries in Cellular Automata

Mohammad Ali Javaheri Javid*§, Mohammad Majid al-Rifaie†§, Robert Zimmer‡§,
§Department of Computing, Goldsmiths, University of London
London SE14 6NW, United Kingdom
*m.javaheri@gold.ac.uk
†m.majid@gold.ac.uk
‡r.zimmer@gold.ac.uk

***Abstract*—In late 1940's and with the introduction of cellular automata, various types of problems in computer science and other multidisciplinary fields have started utilising this new technique. The generative capabilities of cellular automata have been used for simulating various natural, physical and chemical phenomena. Aside from these applications, the lattice grid of cellular automata has been providing a by-product interface to generate graphical patterns for digital art creation. One notable aspect of cellular automata is symmetry, detecting of which is often a difficult task and computationally expensive. This paper uses a swarm intelligence algorithm – Stochastic Diffusion Search – to extend and generalise previous works and detect partial symmetries in cellular automata generated patterns. The newly proposed technique tailored to address the spatially-independent symmetry problem is also capable of identifying the absolute point of symmetry (where symmetry holds from all perspectives) in a given pattern. Therefore, along with partially symmetric areas, the centre of symmetry is highlighted through the convergence of the agents of the swarm intelligence algorithm. This technique is potentially applicable in the domain of aesthetic evaluation where symmetry is one of the measures.**

***Keywords*—Swarm Intelligence; Stochastic Diffusion Search; Symmetry; Cellular automata; Computational Aesthetics**

## I. Introduction

Creating aesthetically pleasing images has been investigated by many researches in the context of evolutionary computing, including the Bimorphs of Dawkins [1], Mutator of Latham [2], and Virtual Creatures of Sims [3]. Although some impressive results have been achieved, there still remains problems with the aesthetic selection. According to [4], first, the subjective comparison process, even for a small number of phenotypes, is slow and forms a bottleneck in the evolutionary process. Human users would take hours to evaluate many successive generations that in an automated system could be performed in a matter of seconds. Secondly, genotype-phenotype mappings are often not linear or uniform. That is, a minor change in genotype may produce a radical change in phenotype. Such non-uniformities are particularly common in tree or graph based genotype representations such as in evolutionary programming, where changes to nodes can have a radical effect on the resultant phenotype. In this study we approach the problem in the framework of dynamical systems and define a criterion for aesthetic selection in terms of its association with symmetry. The association of aesthetics and symmetry has been investigated from different points of view.

In this work, a brief account on cellular automata is presented, followed by a section on symmetry and its significance in aesthetics. Then a swarm intelligence algorithm – Stochastic Diffusion Search – is explained, highlighting its main features, including its unique partial function evaluation aspect. Afterwards, the application of the algorithm in detecting points of symmetry is detailed, illustrating the performance of the method proposed.

## II. Cellular Automata

A cellular automaton is a lattice of regularly arranged homogeneous deterministic finite state automata in Euclidean space.

Therefore, a cellular automaton can be presented as a quadruple of $\mathcal{A} = \langle L, S, N, f \rangle$, where $L$ is a finite square *lattice* in $\mathbb{Z}^2$ with *periodic boundary conditions*, $S \subseteq \mathbb{N}^0$ is a finite set of non-negative integers as *alphabet* or *states* ($S = \{s_0, .., s_{n-1}\}$), $N \subseteq \mathbb{N}^+$ is a finite set of non-negative integers as *neighbourhood* and $f : S^{|N|} \mapsto S$ is a mapping as the *state transition function*. In a 2D lattice with square cells as primitive units if automata on the opposite sides of the lattice (up and down with left and right) are in neighbourhood relation, the resulting lattice forms a virtual torus shape (Fig. 1) which is referred as a lattice with *periodic boundary conditions*. The state transition function (local rule) $f$ maps from the set of neighbourhood states $S^{|N|}$ where $|N|$ is the cardinality of neighbourhood set, to the set of states $S$ *synchronously* at *discrete time* intervals of $t = \{0, 1, 2, 3, ..., n\}$ where $t = 0$ is the *initial time* of a cellular automaton. A mapping that satisfies $f(s_0, ..., s_0) = s_0$ ($s_0 \in S$) is called a *quiescent state*.

The behaviour of CA at a certain point of time emergences from a *synchronous* iterative application of transition function (local rule) over the initial configuration at time $t = 0$. There are some distinctive characteristics in CA which can make them particularly attractive to digital artists and suitable for image and pattern generation purposes (each automaton acting as picture element). Furthermore, the significance of CA for computer art comes from the fact that simple rules can generate

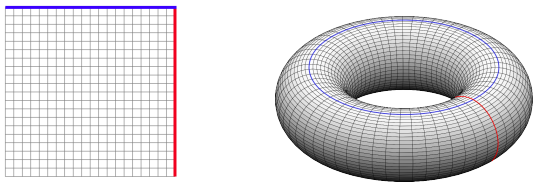


Fig. 1. The formation of periodic boundary from a lattice

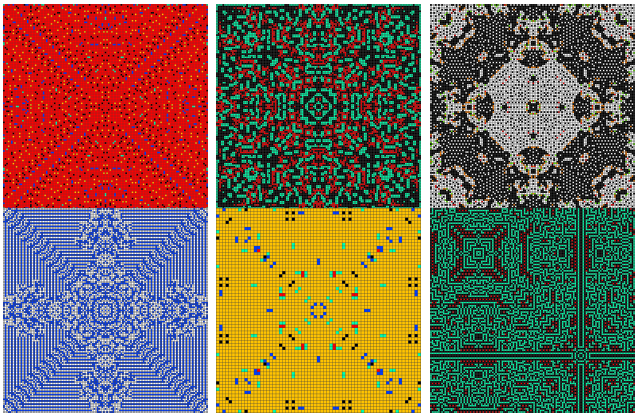


Fig. 2. Sample CA generated images with symmetrical patterns

observationally unpredictable complex behaviours and there is a vast universe of behaviours which can be explored. Generally the behaviour of a particular cellular automaton is constrained by its initial configuration, transaction function and number of states. A two-dimensional multi-state cellular automaton with periodic boundary provides an endless environment for the growth of patterns and the observation of emergent complex behaviour over the time of evolution. For some rules the periodic generation of patterns creates an animated sequence of pattern formations. This opens up possibility of generating animations based on the development of pattern formation where both symmetries and the element of surprise coexist. This capability was observed in [5] where CA are described as "self-generating computer graphics movies". This is a new way of generating imagery which has no precedent in human culture [6]. The role of symmetry in art, architecture and its association with aesthetic preferences is a well known concept [7]. The iterative application of transition function over initial configuration, especially in multi-state CA, can generate complex symmetrical patterns [8], [9] which are extremely challenging to construct using conventional mathematical methods. Fig. 2 shows experimental patterns generated by the authors to demonstrate the generative capabilities of CA in creating symmetrical patterns.

## III. Symmetry and Aesthetic

Symmetry or having proportionality and balance is an important element of aesthetics. The association of aesthetics and symmetry has been investigated extensively in the literature. A study to investigate the effect of symmetry on interface judgements, and relationship between a higher symmetry value and aesthetic appeal for the basic imagery, showed that subjects preferred symmetric over non-symmetric images [10]. Further studies found that if symmetry is present in the face or the body, an individual is judged as being relatively more attractive and if the body is asymmetric the face is rated unattractive, even if the person doing the rating never sees the body [11], [12]. Symmetry plays a crucial role in theories of perception and is even considered a fundamental structuring principle of cognition [13]. In the Gestalt school of psychology things [objects] are affected by where they are and by what surrounds them... so that things [objects] are better described as more than the sum of their parts [14].

The Gestalt principles emphasise the holistic nature of perception where recognition is inferred, during visual perception, more by the properties of an image as a whole, rather than its individual parts [15]. Thus, during the recognition process elements in an image are grouped from parts to whole based on Gestalt principles of perception such as proximity, parallelism, closure, symmetry, and continuation [16]. In particular, symmetric objects are more readily perceived [17]. It is not surprising that we humans find sensory delight in symmetry, given the world in which we evolved. In our world the animals that have interested us and our ancestors (as prey, menace, or mate) are overwhelming symmetric along at least one axis [18].

### A. Evolutionary and Computational approaches

Evolutionary psychologists examine physical appearances like as symmetry, and perceived level of aesthetics as an indirect measure in mate selection [7], [19]. In this view symmetrical faces are examined as more attractive faces. In other words symmetry is positively linked with both psychological and physiological health indicators [20]. In geometry symmetrical shapes are produced by applying four operations of translations, rotations, reflections, and glide reflections. However developing computational methods which generate symmetrical patterns is still a challenge since it has to connect abstract mathematics with the noisy, imperfect, real world; and few computational tools exist for dealing with real-world symmetries [21]. Applying evolutionary algorithms to produce symmetrical forms leaves the formulation of fitness functions, which generate and select symmetrical phenotypes, to be addressed. Lewis describes two strategies in evolutionary algorithms approach for generating and selecting symmetrical forms: "A common approach is to hope for properties like symmetry to gradually emerge by selecting for them. Another strategy is to build in symmetry functions which sometimes activate, appearing suddenly. However this leads to a lack of control, as offspring resulting from slight mutations (i.e., small steps in the solution space) bear little resemblance to their ancestors [22]".

There are several algorithms designed to detect exact symmetry in images; amongst which, and in the case of a collection of $n$ points in the plane, Atallah describes an algorithm for enumerating all axes of symmetry under reflection of a planar shape [23]. In another work, Wolter et al. give exact algorithms, based on string matching, for the detection of symmetries of point clouds, polygons, and polyhedra [24]. These algorithms are often impractical due to their sensitivity to noise and high computational expense, because of their restricted nature to exact symmetries.

In terms of approximate symmetry there are two broad categories: the first defines approximate symmetry by an infimum of a continuous distance function quantifying how similar is a

shape to its transformed version. One of the works introduced in this category is reported in [25]. The second approach aims to address the computational complexity issue by translating the search into a proxy domain, realising that the set of admissible symmetries is sparse in the transformation space; among the examples of this type of work are [26], [27], [28]. Other variations of symmetry detection have being proposed that fall within the broad above-mentioned categories (e.g. [29] which addresses reflection symmetry detection problem or [30] focusing on boolean functions, which is based on Shannons Theorem [31] that detects symmetry by comparison of two-variable cofactors).

In the research presented in this paper, two modalities of the same algorithms are used to detect absolute symmetry in an input image if present, as well partial (sub-) symmetries.

The next section explains the swarm intelligence algorithm which will be used in detecting symmetrical patterns.

## IV. Stochastic Diffusion Search

The swarm intelligence algorithm used in this work is Stochastic Diffusion Search (SDS) [32], [33] which is a probabilistic approach for solving best-fit pattern recognition and matching problems. SDS, as a multi-agent population-based global search and optimisation algorithm, is a distributed mode of computation utilising interaction between simple agents. Its computational roots stem from Geoff Hinton's interest 3D object classification and mapping. See [34] for Hinton's work and [35] for the connection between Hinton mapping and SDS. SDS algorithm has been used in various fields including optimisation, generative arts and medical imaging (e.g. [36], [37], [38], [39]). SDS, contrary to many swarm intelligence algorithms, has a strong mathematical framework describing its behaviour and convergence. The full mathematical model and proof of SDS convergence are elaborated in [33].

### A. SDS Architecture

Similar to other swarm intelligence algorithms, SDS commences a search or optimisation by initialising its population. In any SDS search, each agent maintains a hypothesis, *h*, defining a possible problem solution. After initialisation, the two phases of SDS – Test and Diffusion phases – are followed (see Algorithm 1 for a high-level description of SDS).

In the test phase, SDS checks whether the agent hypothesis is successful or not by performing a partial hypothesis evaluation and returning a domain independent boolean value. Later in the iteration, contingent on the strategy employed, successful hypotheses diffuse across the population and in this way information on potentially good solutions spreads throughout the entire population of agents.

In other words, in the Test phase, each agent performs *partial function evaluation,* $pFE$, which is some function of the agent's hypothesis, $pFE = f(h)$; and in the Diffusion phase, each agent recruits another agent for interaction and potential communication of hypothesis.

### B. Standard SDS and Passive Recruitment

In standard SDS, *passive recruitment mode* is employed. In this mode, if the agent is inactive, a second agent is randomly selected for diffusion; if the second agent is active, its hypothesis is communicated (*diffused*) to the inactive one. Otherwise there is no flow of information between agents; instead a completely new hypothesis is generated for the first inactive agent at random (see Algorithm 2). Therefore, recruitment is not the responsibility of the active agents. Higher rate of inactivity boosts exploration, whereas a lower rate biases the performance towards exploitation. Details of the test phase and the fitness function is described later in this paper.

**Algorithm 1** SDS Algorithm

```
01: Initialising agents
02: While (stopping condition is not met)
03:    Testing hypotheses
04:       Determining agents status (active/inactive)
05:    Diffusing hypotheses
06:       Exchanging of information
07: End While
```

**Algorithm 2** Passive Recruitment Mode

```
01: For each agent ag
02:    If ( !ag.isActive )
03:       r_ag = pick a random agent
04:       If ( r_ag.isActive )
05:          ag.hypothesis = r_ag.hypothesis
06:       Else
07:          ag.hypothesis = generate random hypothesis
08:    End If
09: End For
```

### C. Partial Function Evaluation

One of the concerns associated with many optimisation algorithms (e.g. Genetic Algorithm, Particle Swarm Optimisation and etc.) is the repetitive evaluation of a computationally expensive fitness functions. In some applications, such as tracking a rapidly moving object or generation of CA patters, the repetitive function evaluation significantly increases the computational cost of the algorithm. Therefore, in addition to reducing the number of function evaluations, other measures can be used in an attempt to reduce the computations carried out during the evaluation of each possible solution, as part of the overall optimisation (or search) processes.

The commonly used benchmarks for evaluating the performance of swarm intelligence algorithms are typically small in terms of their objective functions computational costs [40], [41], which is often not the case in real-world applications (examples of costly evaluation functions are seismic data interpretation, selection of sites for the transmission infrastructure of wireless communication networks and radio wave propagation calculations of one site, etc.).

Costly objective function evaluations have been investigated under different conditions [42] and the following two broad approaches have been proposed to reduce the cost of function evaluations:

- The first is to estimate the fitness by taking into account the fitness of the neighbouring elements, the former generations or the fitness of the same element through statistical techniques introduced in [43].

- In the second approach, the costly fitness function is substituted with a cheaper, approximate fitness function.

When agents are about to converge, the original fitness function can be used for evaluation to check the validity of the convergence [42].

The approach that the standard SDS algorithm uses is similar to the second method. Many fitness functions are decomposable to components that can be evaluated separately. During the test phase of SDS, in partial function evaluation ($pFE$, which is some function of the agent's hypothesis, $pFE = f(h)$), the evaluation of one or more of the components may provide partial information to guide the subsequent optimisation process.

In other words, instead of evaluating the hypothesis in its entirely, part of it, which is called *micro-feature*, is selected and evaluated accordingly. Therefore, during the test phase, only the randomly selected micro-features of the hypotheses are evaluated and the status of each agent is thus determined. Thus, if the micro-feature of each hypothesis consists of, say, $\frac{1}{10}$ of the entire hypothesis, the computational expense for the evaluation process of each hypothesis would be $\frac{9}{10}$ computationally cheaper.

Next, details of the process through which SDS performs its spatial-independent symmetry detection is presented.

## V. Experiments

This section explains the design of the experiments conducted along with the results of applying SDS to identify partial or full symmetries on the cellular automata generated patterns. The inputs to the system are sample patterns used as proof of principle to show the functionality of the method; afterwards, some real world cellular automata generated patterns are fed in the system to evaluate the overall performance of the algorithm in detecting the aforementioned types of symmetries.

In order to adopt SDS to identify symmetries, the following important considerations are taken into account:

- the search space comprises of the entire cells on the grid
- SDS hypothesis is a cell index. For instance, in a $5 \times 5$ grid, the coordinate $(2, 2)$ could be the hypothesis and micro-features[1] can be selected by specifying the $x_d$ and $y_d$ distances from the hypothesis; therefore, assuming the $(x_d, y_d)$ distance is $(2, 0)$, this micro-feature should be compared against its corresponding element with $(-2, 0)$ distance from the hypothesis.
- the environment in cellular automata is torus, which means if moving downwards along the search space when we reach the last raw, the next row to be visited is the top row. The same is applicable when moving between columns (see Fig. 1 shows the 2D representation of the cellular automata and its real structure as torus).

[1]Micro-features are used in the test phase of SDS to determine the status of the agent (i.e. active or inactive).

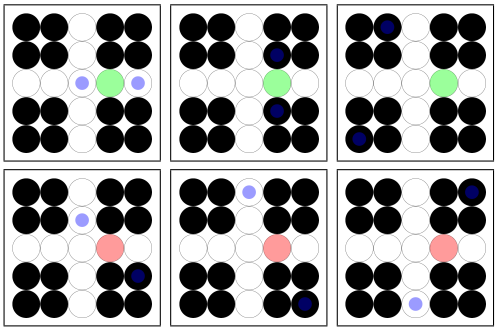


Fig. 3. Sample hypothesis set to be $(3, 2)$; active hypotheses are shown in green and the inactive ones are displayed in red; the selected micro-features are highlighted in blue

The patterns in Figs. 3 show the hypothesis $(3, 2)$ and the various possible micro-features, some of which resulting in the hypothesis' status to be *true* while some others lead to the hypothesis' status to be *false*. The hypothesis in these figures are set to be $(3, 2)$ and various micro-features are selected to test the symmetry of the pattern along various axes of symmetry. The torus structure of cellular automata is demonstrated in the choice of some of the corresponding micro-features; see, for example, Fig. 3 top-right corner, where the micro-feature is chosen at $(-1, -1)$ distance. Thus the corresponding cell is chosen at $(1, 1)$ distance from the hypothesis, which means moving out of the 2D canvas from the right border and entering again from the left.

The process through which SDS commences with the initialisation phase and then cycle through the two phases and test and diffusion is explained next.

### A. Initialisation phase

During the initialisation phase each one of the agents in the population is assigned a hypothesis which is a random $(x, y)$ coordinate from the search space. Additionally, the status of all agents are initially set to *false*.

### B. Test phase

In the test phase, each agent, which is already allocated an $(x, y)$ hypothesis, picks a random $x_d$ and $y_d$ distances from the hypothesis cell as its micro-feature; the randomly selected micro-feature is then compared against corresponding mirrored cells to checks if the mirror cell has the same value. If the values are the same, the status of the agent is set to *true*, otherwise *false*

### C. Diffusion phase

The process in the diffusion phase is the same as the one detailed in the algorithm description where each inactive agent picks an agent randomly from the population; if the randomly selected agent is active, the inactive agent adopts the hypothesis of the active agent (i.e. the $(x, y)$ coordinate), otherwise the inactive agent picks a random coordinate from the search space.

After $n$ number of iterations agents converge on the $(x, y)$ coordinates with the most symmetrical quality.

### D. Experiments and Discussion

One of the main features of SDS is partial function evaluation which here manifests itself in: each time comparing one cell on one side of the symmetrical point to its corresponding cell on the other side. Therefore even when an agent is active, in the next iteration it picks another micro-feature and checks the point from "a different perspective" to ensure that the symmetry still holds. In other words, using this approach, the algorithm allocate its resources "wisely" and repeatedly tests the already maintained points of interest against any asymmetrical discovery.

For the experiments reported in this work, the population size is empirically calculated using the following formula:

$$\text{pSize} = \frac{w^2}{4} \tag{1}$$

where pSize is the population size and $w$ is the width of the search space. Using this set-up, the agents land on fourth of the search space; therefore for a $5 \times 5$ search space, $pSize = 6$.

As illustrated in the figures, some agents became active on $(x, y)$ coordinates which do not represent the full four-fold symmetry; these agents will eventually pick different micro-features in the next iterations and become inactive; consequently, when they are inactive, they need to choose random agents; given that the number of active agents on the centre of symmetry increases over time (thanks to the diffusion phase), it is likely that an active agent is chosen. This would lead to the inactive agents picking micro-features from the centre of symmetry in their next iterations and become/stay active. Note that in these experiments, alpha is used for the transparency of the agents' colour; therefore as shown on the figures, the cell with the largest number of active agents can be distinguished from others.

There are occasions when more than one centre of symmetry exists, or there exist some partial symmetries (also called sub-symmetry) in the image along with full centre of symmetry; in this case another flavour of the recruitment strategy is deployed which is called *context-sensitive mechanism*. This strategy frees up some of the agents who are active and share the same hypothesis, and therefore allows the algorithm to constantly check for traces of symmetry in the input pattern.

**Algorithm 3** Context Sensitive Mechanism

```
01: If ( ag.activity )
02:    r_ag = pick a random agent
03:    If ( r_ag.activity AND
04:         ag.getHypothsis == r_ag.getHypothsis )
05:       ag.setActivity ( false )
06:       ag.setHypotheis ( randomHypothsis )
07:    End If
08: End If
```

In other words, the use of context sensitive mechanism biases the search towards global exploration. Thus, if an active agent randomly chooses another active agent that maintains the same hypothesis, the selecting agent is set inactive and adopts a random hypothesis. This mechanism frees up some of the resources in order to have a wider exploration throughout the search space as well as preventing cluster size from overgrowing; this process goes on while ensuring the formation of large clusters in case there exists a perfect match or good sub-optimal solutions (see Algorithm 3).

The next set of experiments use some complex patterns, generated by cellular automata techniques. Initially an experiment is run that utilises the *passive recruitment mode* without the introduced *context-sensitive mechanism* and later, the impact of *context-sensitivity* is discussed.

The graph in Fig. 4 illustrate the behaviour of the agents' activities; this graph demonstrates that after the initialisation phase, the number of active and inactive agents are balanced; however over time, and due to the presence of a centre of symmetry in the pattern, the number of active agents increases and the number of inactive agents decreases. Therefore, ultimately, once the absolute center of symmetry (where symmetry holds irrespective of which micro-feature is chosen) is identified, the entire agent population becomes active and the number of inactive agents drops to zero.

Using context-sensitive mechanism, the graph in Fig. 5 illustrates the behaviour of SDS algorithm using this mode, where the populations are biased towards global exploration. In this graph, while the increase of active agents and the decrease of inactive agents are visible, it is evident that there are always agents which are released back from the centre of symmetry to the search space to explore the possibility of the *presence of further (sub-) symmetrical points*. This feature is particularly useful in dynamic environment, and where there are more than one absolute points of symmetry (the next experiment uses an input image with a few points of symmetries). The figure shows many active (green) and inactive (red) agents throughout the search space and the graph illustrates that the number of inactive agents never drops to zero.

The next experiment, which uses a more symmetrically complex CA-generated pattern, demonstrates the crucial difference between using SDS with and without the context-sensitive mechanism. As stated before, context-sensitive mechanism reduces the greediness of the agents and allows the agents to explore the search space for any undetected symmetry, while the pure passive mechanism is greedy and once it finds the absolute point of symmetry (where symmetry holds no matter which micro-feature is picked), it gradually pulls all the agents towards the point and stops them from locating possible partial symmetries in the canvas.

The new input to be used in this experiment has two identically CA grown patterns one on the top-left corner and another on on the bottom-left corner. When running the SDS algorithm, it becomes clear that the passive recruitment strategy (see Fig. 6) initially locates two points of symmetry (at $n = 100$ iterations), however later (at $n = 200$ iterations) all agents are drawn towards the absolute point of symmetry (note that the search spaces in cellular automata are torus).

Using the context sensitive approach, the largest partial symmetries are also identified and highlighted (see Fig. 7). The graphs at the bottom of Figs. 6 and 7 clearly show the behaviour of the agents in both modes. As displayed in the graph of Fig. 7, while the number of active and inactive agents are distinguishably far from one another, yet it is shown that the number of active agents does not reach the maximum

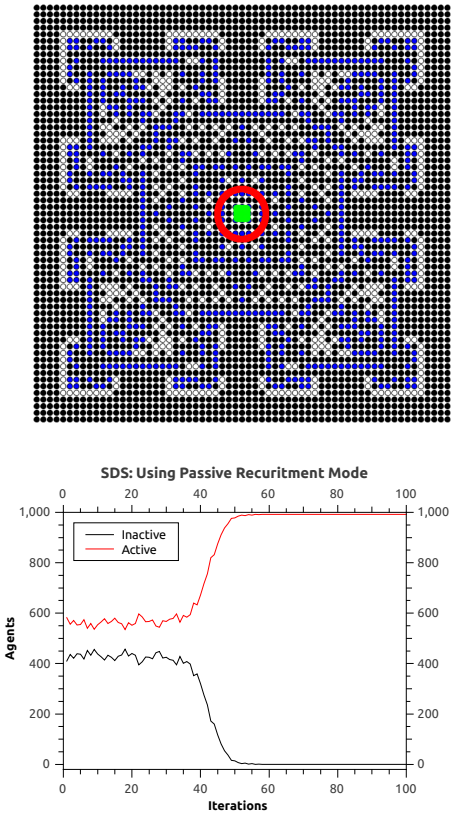



Fig. 4. Passive recruitment mode

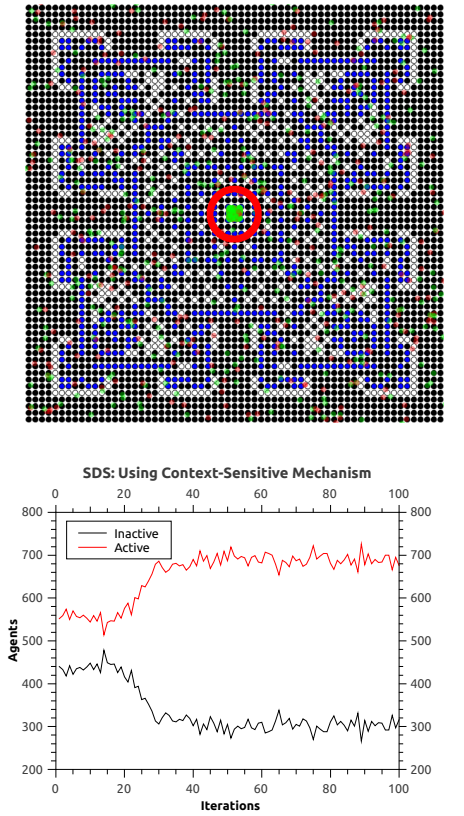



Fig. 5. Context sensitive mechanism

possible[2] and the number of inactive agents does not drop to zero. This mechanism insures the identification of other (sub-) symmetrical points in the input. Therefore, depending on the functionalities needed, either of these approaches could be used.

Another observation to be expanded in the future work is the direct proportionality of the agents' activity to the 'strength' of the symmetry. Therefore, while context-sensitive mechanism finds partial symmetries, it is able to '*rank*' the various clusters of agents which are formed over the pattern. This could lead to introducing the ratio of active/inactive agents as a measure for $order$ and $complexity$ along with Shannon's entropy [44] and information gain [45], [46], [47] which are among the very few measures used in cellular automata for measuring symmetry.

## VI. Conclusion

CA provide perspective and powerful tools in generating computer graphics. The multi-state CA rule space is a vast set of possible rules which can generate interesting patterns with high aesthetic qualities. The interaction of CA rules at local level generates emergent global behaviour, that can sometimes demonstrate attractive complexity. Some characteristics of CA, such as the regularity and complexity of the rules that are employed locally, suggest that they could be well suited to generating computer graphics.

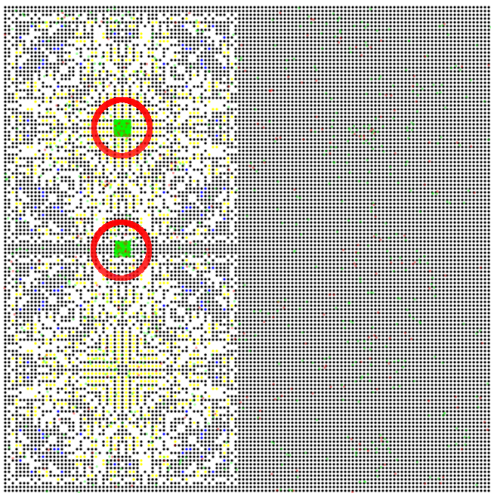
This paper demonstrates the capability of a swarm intelligence algorithm – Stochastic Diffusion Search – in detecting absolute symmetries (when present) and the centre of partial symmetrical patterns within the input image. Evaluating the symmetry of cellular automata generated patterns is often a difficult task partly due the the large size of the search space, and partly due to the constantly changing, dynamic environment in which the cellular automata patterns are generated. These factors contribute to making the detection of symmetrical patterns computationally expensive. One of the main features of Stochastic Diffusion Search is partial function evaluation which is particularly useful when dealing with large problems with high dimensions and costly evaluation function (e.g. in this case, the expensive computational cost of detecting symmetry in cellular automata generated patters). The performance of this algorithm is explained in the paper and the results are accordingly demonstrated.

Following the introduction of this novel technique, among the future research topics are: conducting a comparison with other evolutionary and non-evolutionary techniques, computing the correlation between the size of search space and the computational complexity of the process, ranking the quality of the symmetries detected, and applying this method to dynamically evolving cellular automata generated patterns.

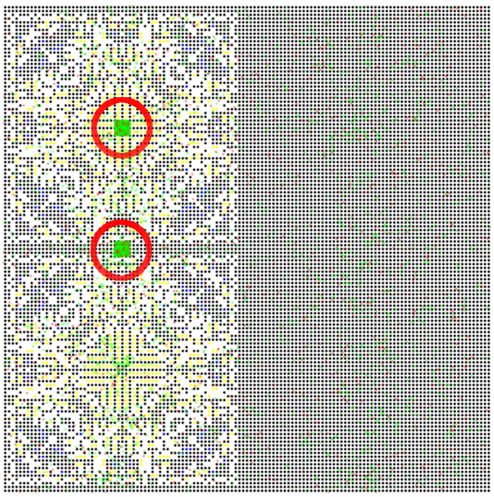
[2]Given the size of the side of search space is $ssSize = 129$, the population size for this pattern is $pSize = \frac{129^2}{4} = 4,160$

## References

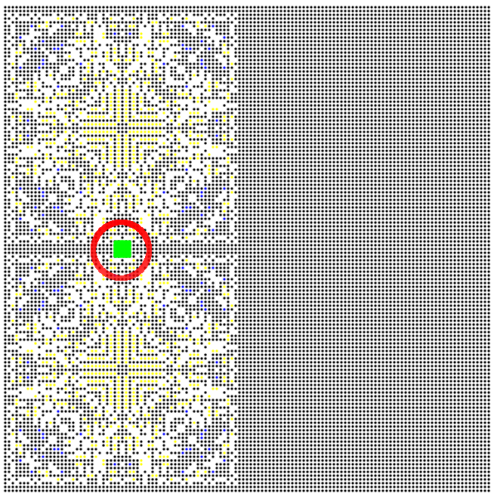
[1] R. Dawkins, *The blind watchmaker*. New York: Norton & Company, Inc, 1986.



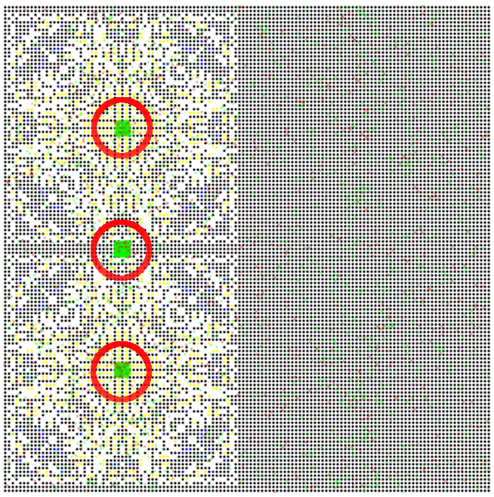
$n = 100$ iterations



$n = 100$ iterations



$n = 200$ iterations



$n = 200$ iterations

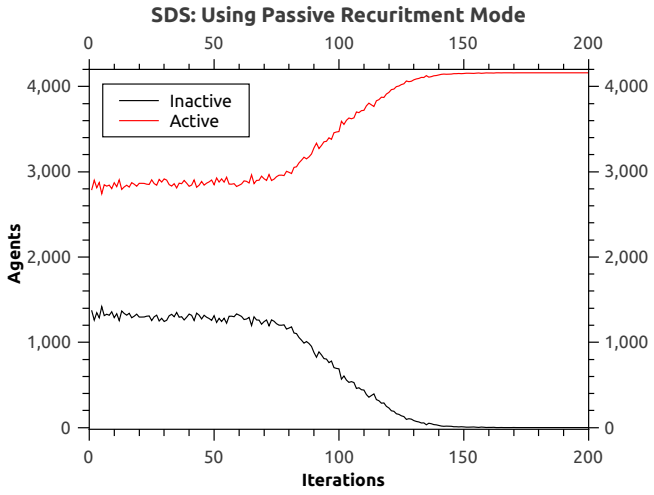



Fig. 6. Passive Recruitment mode: finding absolute symmetry

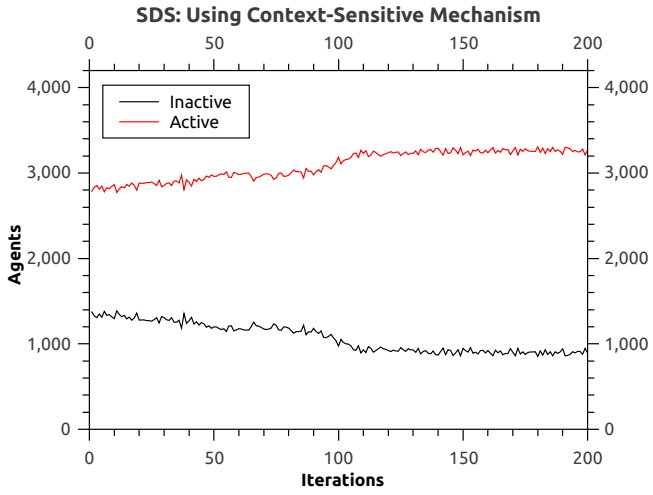



Fig. 7. Context-Sensitive mechanism: finding partial symmetry

[2] S. Todd, W. Latham, and P. Hughes, "Computer sculpture design and animation," *The Journal of Visualization and Computer Animation*, vol. 2, no. 3, pp. 98–105, jul–sep 1991.

[3] K. Sims, "Evolving virtual creatures," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. ACM, 1994, pp. 15–22.

[4] J. McCormack, "Interactive evolution of l-system grammars for computer graphics modelling," *Complex Systems: from biology to computation*, pp. 118–130, 1993.

[5] R. Rucker, *Seek!: Selected Nonfiction*. Running Press Book Publishers, 1999.

[6] T. O. Roth and A. Deutsch, "Universal synthesizer and window: Cellular automata as a new kind of cybernetic image," in *Imagery in the 21st Century*. The MIT Press, 2011, pp. 269–288.

[7] P. Møller A. and T. R., "Bilateral symmetry and sexual selection: a meta-analysis," *Am. Nat*, vol. 151, no. 2, pp. 174–192, 1998.

[8] M. A. J. Javid and R. te Boekhorst, "Cell Dormancy in Cellular Automata," in *International Conference on Computational Science (3)*, ser. Lecture Notes in Computer Science, V. N. Alexandrov, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, Eds., vol. 3993. Springer, 2006, pp. 367–374.

[9] M. A. Nowak, *Evolutionary dynamics: exploring the equations of life*. Harvard University Press, 2006.

[10] M. Bauerly and Y. Liu, "Computational modeling and experimental investigation of effects of compositional elements on interface and design aesthetics," *International Journal of Man-Machine Studies*, vol. 64, no. 8, pp. 670–682, 2006.

[11] T. Randy and G. Steven, "Human facial beauty," *Human Nature*, vol. 4, pp. 237–269, 1993.

[12] S. W. Gangestad, R. Thornhill, and R. A. Yeo, "Facial attractiveness, developmental stability, and fluctuating asymmetry," *Ethology and Sociobiology*, vol. 15, no. 2, pp. 73–85, 1994.

[13] M. Leyton, *Symmetry, causality, mind*, ser. Bradford Books. MIT Press, 1992.

[14] R. Behrens, *Design in the visual arts*. Prentice-Hall, 1984.

[15] H. Jiang, C. W. Ngo, and H. K. Tan, "Gestalt-based feature similarity measure in trademark database," *Pattern Recognition*, vol. 39, no. 5, pp. 988–1001, May 2006.

[16] I. K. Park, K. M. Lee, and S. U. Lee, "Perceptual grouping of line features in 3-D space: A model-based framework," *Pattern Recognition*, vol. 37, no. 1, pp. 145–159, Jan. 2004.

[17] Carroll and J. M., Eds., *HCI Models, Theories, and Frameworks: Toward a multidisciplinary science*. San Francisco: Morgan Kaufmann Publishers, 2003.

[18] P. Railton, *Aesthetic Value, Moral Value and the Ambitions of Naturalism In Aesthetics and Ethics*. University of Maryland, 2001, ch. 3.

[19] P. Møller A. and J. J. Cuervo, *Asymmetry, size and sexual selection : meta-analysis, publication bias and factors affecting variation in relationships*. Oxford University Press, 1999, p. 1.

[20] T. K. . L. R. J. Shackelford, "Facial symmetry as an indicator of psychological emotional and physiological distress," *Journal of Personality and Social Psychology*, vol. 72, 1997.

[21] Y. Liu, "Computational symmetry," in *CMU Robotics Institute*, 2000.

[22] M. Lewis, "Evolutionary visual art and design," in *The Art of Artificial Evolution*, ser. Natural Computing Series, J. Romero and P. Machado, Eds. Springer, 2008, pp. 3–37.

[23] M. J. Atallah, "On symmetry detection," *Computers, IEEE Transactions on*, vol. 100, no. 7, pp. 663–666, 1985.

[24] J. D. Wolter, T. C. Woo, and R. A. Volz, "Optimal algorithms for symmetry detection in two and three dimensions," *The Visual Computer*, vol. 1, no. 1, pp. 37–48, 1985.

[25] H. Zabrodsky, S. Peleg, and D. Avnir, "Symmetry as a continuous feature," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 17, no. 12, pp. 1154–1166, 1995.

[26] C. Sun and J. Sherrah, "3d symmetry detection using the extended gaussian image," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 2, pp. 164–168, 1997.

[27] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser, "A planar-reflective symmetry transform for 3d shapes," in *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3. ACM, 2006, pp. 549–559.

[28] N. J. Mitra, L. J. Guibas, and M. Pauly, "Partial and approximate symmetry detection for 3d geometry," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 560–568, 2006.

[29] S. Lee and Y. Liu, "Curved glide-reflection symmetry detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 2, pp. 266–278, 2012.

[30] J. S. Zhang, M. Chrzanowska-Jeske, A. Mishchenko, and J. R. Burch, "Generalized symmetries in boolean functions: Fast computation and application to boolean matching," in *in IWLS*. Citeseer, 2004.

[31] C. Shannon *et al.*, "The synthesis of two-terminal switching circuits," *Bell System Technical Journal*, vol. 28, no. 1, pp. 59–98, 1949.

[32] J. Bishop, "Stochastic searching networks," in *Proc. 1st IEE Conf. on Artificial Neural Networks*, London, UK, 1989, pp. 329–331.

[33] M. M. al-Rifaie and M. Bishop, "Stochastic diffusion search review," in *Paladyn, Journal of Behavioral Robotics*. Paladyn, Journal of Behavioral Robotics, 2013, vol. 4(3), pp. 155–173. [Online]. Available: http://dx.doi.org/10.2478/pjbr-2013-0021

[34] G. F. Hinton, "A parallel computation that assigns canonical object-based frames of reference," in *Proceedings of the 7th international joint conference on Artificial intelligence-Volume 2*. Morgan Kaufmann Publishers Inc., 1981, pp. 683–685.

[35] J. Bishop and P. Torr, "The stochastic search network," in *Neural Networks for Images, Speech and Natural Language*, Chapman & Hall, New York, 1992, pp. 370–387.

[36] M. M. al-Rifaie, M. Bishop, and T. Blackwell, "Information sharing impact of stochastic diffusion search on differential evolution algorithm," in *J. Memetic Computing*. Springer-Verlag, 2012, vol. 4, no. 4, pp. 327–338. [Online]. Available: http://dx.doi.org/10.1007/s12293-012-0094-y

[37] M. M. al-Rifaie, M. Bishop, and S. Caines, "Creativity and autonomy in swarm intelligence systems," in *J. Cognitive Computation*. Springer-Verlag, 2012, vol. 4, no. 3, pp. 320–331. [Online]. Available: http://dx.doi.org/10.1007/s12559-012-9130-y

[38] A. M. Al-Rifaie and M. M. al-Rifaie, "Generative music with stochastic diffusion search," in *Evolutionary and Biologically Inspired Music, Sound, Art and Design*, ser. Lecture Notes in Computer Science, C. Johnson, A. Carballal, and J. Correia, Eds. Springer International Publishing, 2015, vol. 9027, pp. 1–14. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-16498-4_1

[39] F. M. Al-Rifaie and M. M. al-Rifaie, "Investigating stochastic diffusion search in dna sequence assembly problem," in *SAI Intelligent Systems Conference 2015*. IEEE, 2015.

[40] J. Digalakis and K. Margaritis, "An experimental study of benchmarking functions for evolutionary algorithms," *International Journal*, vol. 79, pp. 403–416, 2002.

[41] D. Whitley, S. Rana, J. Dzubera, and K. E. Mathias, "Evaluating evolutionary algorithms," *Artificial Intelligence*, vol. 85, no. 1-2, pp. 245–276, 1996.

[42] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *In: Soft Computing*, vol. 9, pp. 3–12, 2005.

[43] J. Branke, C. Schmidt, and H. Schmeck, "Efficient fitness estimation in noisy environments," *In Spector, L., ed.: Genetic and Evolutionary Computation Conference, Morgan Kaufmann*, 2001.

[44] C. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423 & 623–656, Oct. 1948.

[45] J. E. Bates and H. K. Shepard, "Measuring complexity using information fluctuation," *Physics Letters A*, vol. 172, no. 6, pp. 416–425, 1993.

[46] Andrienko, Yu. A., Brilliantov, N. V., and Kurths, J., "Complexity of two-dimensional patterns," *Eur. Phys. J. B*, vol. 15, no. 3, pp. 539–546, 2000.

[47] R. Wackerbauer, A. Witt, H. Atmanspacher, J. Kurths, and H. Scheingraber, "A comparative classification of complexity measures," *Chaos, Solitons & Fractals*, vol. 4, no. 1, pp. 133–173, 1994.