applied sciences

MDPI

Article

Exploration of Swarm Dynamics Emerging from Asymmetry

Naoki Nishikawa *, Reiji Suzuki and Takaya Arita

Department of Complex Systems Science, Graduate School of Information Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-0814, Japan; reiji@nagoya-u.jp (R.S.); arita@nagoya-u.jp (T.A.)

* Correspondence: nishikawa@alife.cs.is.nagoya-u.ac.jp; Tel.: +81-52-789-3503

Received: 5 March 2018; Accepted: 27 April 2018; Published: 5 May 2018

Abstract: A swarm might exhibit interesting motions or structures when it includes different types of agents. On a swarm model named Swarm Chemistry, some interesting patterns can appear if the parameters are well-tuned. However, there is a hurdle for us to get capable of tuning the parameters by automatic searching methods like a genetic algorithm, particularly because defining interestingness itself is a challenging issue. This paper aims to investigate how interesting patterns can be detected, comparing seven measures from an aspect of system asymmetries. Based on numerical experiments, the effects of changing kinetic parameters are discussed, finding that: (1) segregating patterns, which are frequently observed but uninteresting, tend to appear when the perception range is small and normal (ideal) speed is large or when cohesive force is weak and separating force is strong; (2) asymmetry of information transfer represented by topological connectivity is an effective way to characterize the interestingness; (3) pulsation-like patterns can be captured well by using time-derivative of state variables like network-degrees; (4) it helps capture a gradual structural deformation when fitness function adopts the mean over min-max differences of state variables. The findings will help the efficient search of already-discovered or undiscovered interesting swarm dynamics.

Keywords: swarm behavior; Swarm Chemistry; self-organization; asymmetrical interaction; genetic algorithm

1. Introduction

Collective behaviors of swarms can be seen in nature, e.g., ant swarms, birds flocks and fish schools, as well as in human society around us. It is an interesting fact that, without having either a very sophisticated intelligence in each component, nor a centralized mechanism to control them, unexpected patterns or dynamics sometimes appear if once they gather in numbers. Each component member in such swarms usually acts based on some simple recognition, decision and action rules. In this paper, we call such a component an agent.

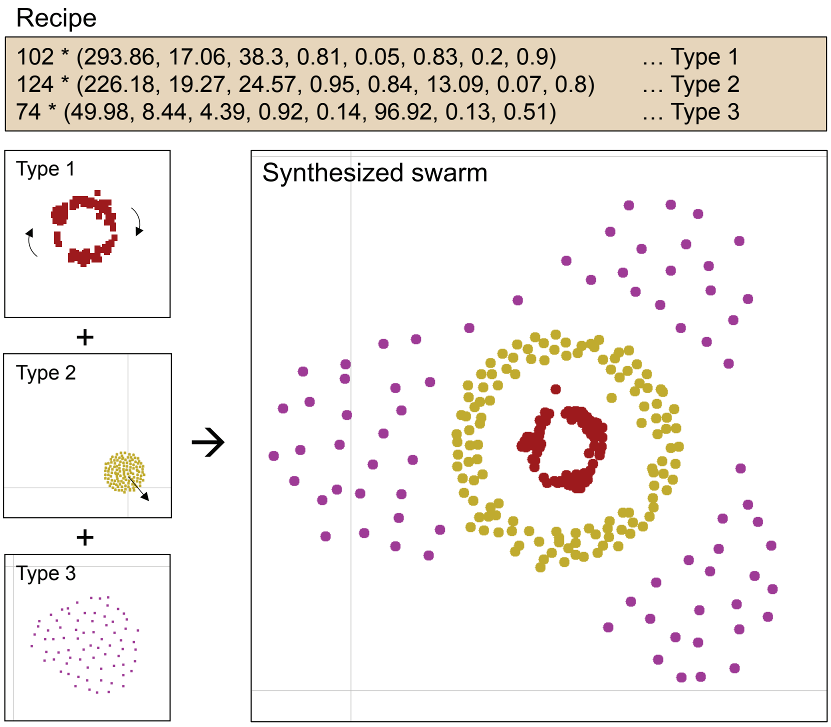
1.1. Swarm Robotics

There have been a lot of work to simulate and analyze swarming behaviors or structures using computational models (e.g., [1–3]). One of the best-known swarm models is Reynolds' Boids [4]. The motions of agents are governed only by three simple interaction rules among agents: cohesion, separation and alignment. In spite of its simplicity of interaction, agents can self-organize into some interesting formation patterns like bird swarms.

From an engineering perspective, there have been many studies in the field of swarm robotics to promote practical applications of swarm behaviors. For example, swarm models can be applied to control design of autonomous robots like unmanned ground vehicles (UGVs), aerial vehicles (UAVs) and marine vehicles (UMVs) exploited in disaster situations [5].

1.2. Heterogeneity of Swarm

What happens if a swarm includes different types of agents within it? It is expected that a swarm might exhibit or form much more interesting behaviors than when it is composed of agents all with the same character. Based on the Boids rules, Sayama introduced some extensions and established a swarm model called Swarm Chemistry (SC) [6–13]. Its most notable feature is that it allows the coexistence of different types of agents in the same swarm. The self-organized patterns or behaviors emerging in this model are pretty attractive. An example of heterogeneous swarm appearing in SC simulator, is shown in Figure 1, which is composed of three different types. In addition, a useful interactive graphical user interface (GUI) tool of SC is open to everyone on Sayama's website [14], where one can try to "design" swarms of his/her own preference. This tool helps users create swarms and adjust parameters. Charming swarm patterns appear when the parameters could be well-adjusted. Some of them are listed in Figure 2. To support the readers' visual understanding, animations of these swarms are prepared as a Supplementary Material, Video S1. The details of Swarm Chemistry will be described in the next section.



Recipe

102 * (293.86, 17.06, 38.3, 0.81, 0.05, 0.83, 0.2, 0.9) … Type 1
124 * (226.18, 19.27, 24.57, 0.95, 0.84, 13.09, 0.07, 0.8) … Type 2
74 * (49.98, 8.44, 4.39, 0.92, 0.14, 96.92, 0.13, 0.51) … Type 3

Figure 1. Example recipe of a swarm composed of three different types of agent.

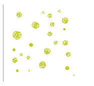
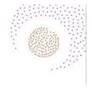
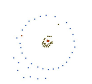
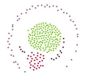
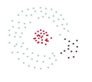
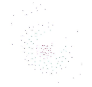
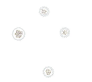
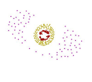
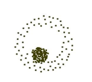
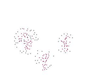
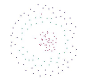
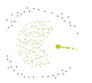
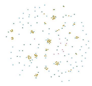
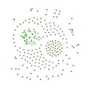
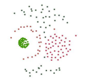
No.	Recipe name	Recipe	Snapshot	Classification of motion
1	Linear Oscillator	133 * (214.41, 17.93, 35.14, 0.64, 0.13, 0.29, 0.08, 0.97) 24 * (253.6, 7.19, 15.51, 0.82, 0.33, 32.65, 0.34, 0.56)		Reciprocating
2	Blobs	300 * (20.8, 1.95, 20.75, 0.95, 0.99, 9.31, 0.05, 0.68)		Clustered
3	Turbulent Runner	131 * (177.1, 9.71, 30.06, 0.8, 0.43, 19.65, 0.45, 0.91) 169 * (277.3, 14.67, 37.71, 0.68, 0.23, 77.01, 0.02, 0.31)		Tracting
4	Wedding Ring	24 * (220.51, 13.88, 3.47, 0.46, 0.38, 6.23, 0.19, 0.68) 13 * (64.07, 1.4, 19.7, 0.88, 0.27, 0.36, 0.47, 0.72) 35 * (117.53, 7.31, 21.72, 0.3, 0.5, 98.69, 0.03, 0.29)		Tracting
5	Rotary	29 * (122.13, 19.19, 17.98, 0.65, 0.44, 19.88, 0.46, 0.2) 51 * (299.13, 0.79, 38.71, 0.25, 0.18, 86.49, 0.38, 0.43) 10 * (252.92, 19.99, 10.21, 0.23, 0.17, 1.22, 0.28, 0.92)		Rotating
6	Swinger	48 * (150.39, 15.89, 23.54, 0.74, 0.45, 62.65, 0.33, 0.13) 152 * (217.14, 12.13, 12.42, 0.59, 0.98, 14.06, 0.04, 0.65) 14 * (248.54, 5.85, 22.26, 0.43, 0.11, 17.14, 0.06, 0.68) 31 * (141.53, 2.91, 4.86, 0.92, 0.03, 21.87, 0.28, 0.2)		Tracting
7	Fast Walker, Slow Follower	67 * (216.35, 11.75, 7.7, 0.83, 0.97, 97.31, 0.02, 0.38) 29 * (254.64, 7.28, 7.0, 0.95, 0.11, 22.41, 0.43, 0.31) 13 * (105.4, 3.55, 5.24, 0.34, 0.18, 23.53, 0.39, 0.24)		Tracting
8	Aggressive Predeator	18 * (211.92, 12.59, 19.37, 0.09, 0.21, 57.92, 0.0, 0.95) 41 * (257.27, 14.96, 35.66, 0.2, 0.8, 47.81, 0.13, 0.13) 35 * (262.68, 2.82, 38.32, 0.21, 0.01, 54.93, 0.11, 0.19) 31 * (78.58, 5.7, 33.23, 0.89, 0.18, 45.44, 0.04, 0.05) 7 * (194.21, 12.88, 21.68, 0.97, 0.19, 99.21, 0.5, 0.13)		Reciprocating
9	Chaos Cells	144 * (109.03, 6.71, 12.7, 0.47, 0.6, 61.43, 0.02, 0.21) 89 * (117.15, 16.33, 31.88, 0.39, 0.13, 12.96, 0.48, 0.8) 67 * (76.3, 8.59, 26.57, 0.7, 0.64, 28.39, 0.3, 0.35)		Clustered
10	Pulsating Eye	102 * (293.86, 17.06, 38.3, 0.81, 0.05, 0.83, 0.2, 0.9) 124 * (226.18, 19.27, 24.57, 0.95, 0.84, 13.09, 0.07, 0.8) 74 * (49.98, 8.44, 4.39, 0.92, 0.14, 96.92, 0.13, 0.51)		Pulsating
11	Playing Catch	76 * (84.06, 0.09, 9.89, 0.33, 0.32, 15.66, 0.22, 0.68) 100 * (158.86, 18.4, 24.98, 0.3, 0.3, 1.72, 0.06, 0.37)		Rotating
12	Recombining Blobs	132 * (45.91, 10.82, 21.11, 0.86, 0.13, 42.48, 0.32, 0.74) 84 * (113.26, 3.41, 25.71, 0.4, 0.39, 49.53, 0.13, 0.24)		Clustered
13	No Wait This Way	60 * (262.68, 2.82, 38.32, 0.21, 0.01, 54.93, 0.11, 0.19) 40 * (78.58, 5.7, 33.23, 0.89, 0.18, 45.44, 0.04, 0.05) 40 * (257.27, 14.96, 35.66, 0.2, 0.8, 47.81, 0.13, 0.13)		Reciprocating
14	Jelly Fish	134 * (262.65, 12.01, 25.87, 0.97, 1.0, 56.35, 0.26, 0.61) 67 * (288.17, 6.19, 23.37, 0.95, 1.0, 1.31, 0.1, 0.9) 68 * (150.5, 12.97, 15.87, 0.46, 0.39, 57.95, 0.17, 0.48)		Tracting
15	Muliticellularity	99 * (19.8, 15.73, 2.61, 0.85, 0.64, 10.51, 0.17, 0.06) 48 * (300.0, 14.63, 0.0, 0.48, 0.81, 90.27, 0.25, 0.78) 37 * (275.18, 16.9, 7.05, 0.48, 0.81, 90.27, 0.17, 0.85) 8 * (159.59, 2.09, 24.19, 0.96, 0.59, 76.03, 0.01, 0.07) 42 * (73.07, 1.82, 2.36, 0.27, 0.61, 40.55, 0.22, 0.86)		Cell-like
16	Cell With Two Nuclei	41 * (249.84, 4.85, 28.73, 0.34, 0.45, 14.44, 0.09, 0.82) 26 * (277.87, 15.02, 35.48, 0.68, 0.05, 82.96, 0.46, 0.9) 30 * (277.87, 15.02, 24.44, 0.68, 0.05, 82.96, 0.43, 0.9) 28 * (110.8, 16.12, 38.6, 0.18, 0.34, 14.3, 0.01, 0.01) 48 * (83.79, 13.29, 7.54, 0.08, 0.79, 1.07, 0.15, 0.45) 74 * (269.64, 6.62, 34.69, 0.36, 0.5, 30.2, 0.03, 0.23)		Cell-like
17	Insurmountable Wall	42 * (52.57, 9.91, 20.42, 0.32, 0.76, 1.8, 0.01, 0.64) 25 * (84.87, 8.82, 24.98, 0.91, 0.44, 40.97, 0.18, 0.6) 45 * (220.42, 4.65, 7.53, 0.96, 0.35, 46.18, 0.25, 1.0) 49 * (279.64, 10.29, 35.95, 0.37, 0.49, 38.09, 0.32, 0.89)		Rotating

Figure 2. Already found interesting patterns (recipe names and values were taken from [14] and snapshots were reproduced by the authors). Classification of motions, which will appear again in Table 3, has been added originally in this paper. For animations of these swarms, see Supplementary Material, Video S1.

1.3. Issues to be Addressed

As stated above, SC is an excellent model. Yet, the potential of the emergence of interesting patterns in SC has not been totally revealed and leveraged. For a more comprehensive exploration of this model, an automated way of searching is necessary because the parameter space is so large that finding patterns manually almost impossible. However, another problem arises. Automatic searching such as genetic algorithm (GA) requires a quantitative measure (a fitness function). As of now, such parameters need to be searched through manual try-and-error depending on user's sense without any objective measures. It is not easy to identify the parameters to create interesting patterns systematically. The main reason is that defining "interestingness" itself is a quite difficult problem.

1.4. Past Approaches

Sayama also tried an evolutionary and automated way of letting interesting patterns appear [13,15,16], by introducing additional rules to the original version of SC, which is called the evolutionary SC (ESC). The major points of ESC model include that [13]:

- an agent has either active or inactive (passive) state, meaning it is moving and has a recipe in it, or it is staying still and has no recipe, respectively.
- a recipe is transmitted from an active agent to another passive agent when these two agents collide, making the latter active.
- the active agent differentiates at random into either type defined in the recipe.
- the direction of recipe transmission is determined by some competition mechanisms.

As a result, in ESC model, the recipe of a swarm can change dynamically and continuously, enabling interesting swarms can appear here and there within the simulation space. This means, a swarm can "evolve" in a single simulation run. However, this method aims to reproduce the evolution process in the natural world, principally for scientific purposes. A comprehensive search using explicit fitness functions is not carried out.

To enable systematic search, Nishikawa et al. proposed a GA-based framework to optimize parameters based on SC [17]. It is for the real-world tasks to be realized by the swarm robots and the fitness functions are comparatively objective. For the case of the present paper, however, we need to quantify how interesting the self-organized swarm is, as a fitness function, which is quite challenging because the recognition, creation and designing of these patterns depend on subjective senses of a human.

Regarding how to measure the interestingness, Sayama has tried to measure the interestingness of swarm using 24 different metrics based on the average or the variation of kinetic outcomes (including agent positions and velocities). and topological outcomes (including number of connected components, average size of connected components, clustering coefficient, and link density) [18,19]. From the Monte Carlo simulation results he demonstrated that the interaction between different types of agents (heterogeneity) helped produce dynamic behaviors. Also, statistically significant differences were detected for most of the outcome variables, especially for topological variables. However, it is still challenging to make a complete answer what is the effective metric.

1.5. Structure of This Paper

This paper aims to investigate how an interesting pattern can be detected, introducing seven explicit metrics from an aspect of system asymmetries. Parameter search is done using GA. Based on numerical experiments, the effects of kinetic parameters and major points to capture characteristic dynamics are discussed.

This paper is composed as follows. In Section 2, the basic description of Swarm Chemistry will be provided. Then Section 3 will present an optimization scheme to search such recipes to generate interesting structures or motion. Next, in Section 4, several possible measures to quantify the swarm structure or dynamics will be presented. Subsequently, Section 5 will show the optimization results obtained from the experiments. After that, Section 6 will give a brief overview on the results and analysis of the especially interesting patterns obtained. Finally, Section 7 will conclude the whole paper.

2. Basic Algorithm of Swarm Chemistry

SC is an extended version of one of the best-known swarm models, Boids [4], in which the motions of agents are controlled only by three simple interaction rules among agents: cohesion, separation and alignment. The entire set of kinetic rules in Boids are:

- Straying If there are no other agents within its local perception range, steer randomly.
- Cohesion (c_1) Steer to move toward the average position of nearby agents.
- Alignment (c_2) Steer toward the average velocity of nearby agents.
- Separation (c_3) Steer to avoid collision with nearby agents.
- Randomness (c_4) Steer randomly with a given probability.
- Self-propulsion (c_5) Approximate its speed to its own normal speed.

Algorithm 1 describes the kinetic rules to govern the movement of agents in SC in more detail [9]. $\vec{x}_i$, $\vec{v}_i$, $\vec{v}_i'$ and $\vec{a}_i$ are the location, the current velocity, the next velocity, and the acceleration of the i-th agent, respectively. r and $r_{\pm p}$ are random numbers in $[0,\ 1)$ and $[-p,\ +p)$, respectively. The interaction properties of an agent are described by 8 kinetic parameters (KPs) denoted as follows.

$$(R,\ V_n,\ V_m,\ c_1,\ c_2,\ c_3,\ c_4,\ c_5) \tag{1}$$

The definitions, units and their possible values are summarized in Table 1. Especially, KP4, KP5 and KP6 are the coefficients for the three effects: cohesion, alignment and separation, respectively. They are used to calculate the acceleration of the agent. So their units should be consistent with that of acceleration, i.e., (Pixel · Step^{-2}). For example, KP4 has a unit of (Step^{-2}), because the first term on the right on Line 8 of Algorithm 1,

$$(\langle\vec{x}\rangle - \vec{x}_i)\,, \tag{2}$$

has a unit of position, i.e., (Pixel).

Table 1. Kinetic parameters (KP) in a recipe to control agent behaviors [9]. Each agent is assigned unique values in its recipe.

KP	Name	Min	Max	Definition	Unit
KP1 (β_1)	R^i	0	300	Radius of local perception range	Pixel
KP2 (β_2)	V_n^i	0	20	Normal speed	Pixel Step^{-1}
KP3 (β_3)	V_m^i	0	40	Maximum speed	Pixel Step^{-1}
KP4 (β_4)	c_1^i	0	1	Coefficient for cohesive effect	Step^{-2}
KP5 (β_5)	c_2^i	0	1	Coefficient for aligning effect	Step^{-1}
KP6 (β_6)	c_3^i	0	100	Coefficient for separating effect	Pixel^2 Step^{-2}
KP7 (β_7)	c_4^i	0	0.5	Probability of random steering	–
KP8 (β_8)	c_5^i	0	1	Tendency of self-propulsion	–

In Boids, all the parameters are shared over all the agents. In SC, unlike Boids, each agent is assigned a unique set of kinetic parameters, which represents its "character". Therefore, a swarm is allowed to have two or more different types of agents. A design parameter of a swarm is described by a recipe (i.e., some sets of KPs). If there are multiple types in a swarm, then it is said to be heterogeneous (otherwise homogeneous). When a swarm has M types agents, the recipe $\mathcal{R}$ for it is written as follows.

$$\mathcal{R} = \begin{cases} N_1 & * \, (R,\ V_n,\ V_m,\ c_1,\ c_2,\ c_3,\ c_4,\ c_5)_1 \\ N_2 & * \, (R,\ V_n,\ V_m,\ c_1,\ c_2,\ c_3,\ c_4,\ c_5)_2 \\ & \vdots \\ N_M & * \, (R,\ V_n,\ V_m,\ c_1,\ c_2,\ c_3,\ c_4,\ c_5)_M, \end{cases} \tag{3}$$

where N_j $(j = 1, \ldots, M)$ at the beginning of each row denotes the number of agents that share the same set of KPs. The parameters newly introduced in SC are:

- R: the perception range to define the size of the neighborhood
- V_n: the ideal speed
- V_m: the maximum speed

It should be noted that the perception range R is a particularly important parameter deciding whether an interaction occurs between an agent and another. Figure 3 illustrates the perception ranges of the agents. Each one of them has its perception range R_i. An agent i $(i = 1, \cdots, N)$ perceives and communicates locally with its neighbors when they come within the perception range of agent i. Figure 3b represents the "cohesion" effect, i.e., every agent makes its orientation towards the center position among its neighboring agents. Figure 3c represents the "aligning" effect, i.e., every agent adjusts its velocity (orientation) to the velocity averaged over those of its neighboring agents. Figure 3d represents the "separation" effect to avoid collisions between the agents. The smaller the distance between the agents i and j becomes, the greater the separating (repulsion) force becomes. These effects are involved together and used to determine the acceleration of the agent i as indicated in Line 8 of Algorithm 1.

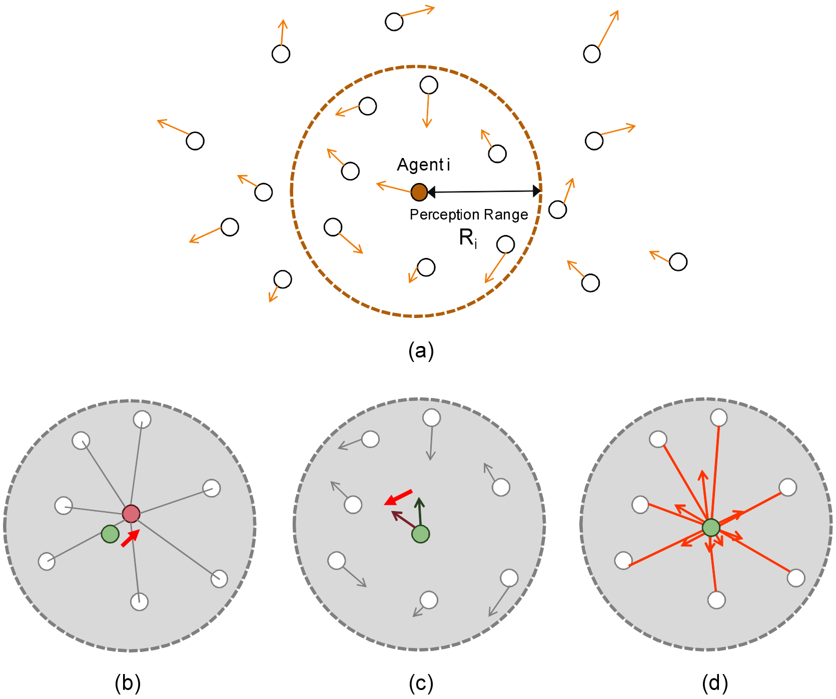


Figure 3. Kinetic interaction between agents (drawn based on [9]). (**a**) Agent i perceives the positions and the velocities of neighboring agents within its perception range, R_i; (**b**) Cohesion; (**c**) Alignment and (**d**) Separation.

Algorithm 1 Kinetic Rules in Swarm Chemistry [9]. Line 2: The set of neighboring agents (i.e., other agents within the local perception range), $\mathcal{N}$ of agent i is found. Lines 3–4: The number of neighboring agents, $|\mathcal{N}|$, is then calculated. If the set $\mathcal{N}$ is empty, i.e., if there is no agent within its perception range, a random acceleration is set, which means a random straying by the agent. Lines 6–7: Otherwise, the average positions and velocities of nearby particles are calculated, respectively. Line 8: A tentative acceleration is calculated according to the three Boids rules (cohesion, alignment and separation). Lines 9–11: At a given probability c_4, a random perturbation is applied to the tentative acceleration. Line 13: A tentative velocity is calculated by time-integrating the acceleration. Line 14: The tentative velocity is limited to prevent overspeed. Line 15: The speed is approximated to its own normal speed (self-propulsion). Lines 17–20: After the above calculation is complete for all the agents, the velocities and then the locations are updated by time-integration.

```
1:  for all $i \in$ agents do
2:    $\mathcal{N} \leftarrow \{j \neq i\}$ that satisfies $|\vec{x}_j - \vec{x}_i| < R^i$
3:    if $|\mathcal{N}| = 0$ then
4:      $\vec{a} \leftarrow (r_{\pm 0.5},\ r_{\pm 0.5})$ // Random steering
5:    else
6:      $\langle\vec{x}\rangle \leftarrow \sum_{j\in\mathcal{N}} \vec{x}_j / |\mathcal{N}|$
7:      $\langle\vec{v}\rangle \leftarrow \sum_{j\in\mathcal{N}} \vec{v}_j / |\mathcal{N}|$
8:      $\vec{a} \leftarrow c_1^i \left(\langle\vec{x}\rangle - \vec{x}_i\right) + c_2^i \left(\langle\vec{v}\rangle - \vec{v}_i\right) + c_3^i \sum_{j\in\mathcal{N}} \left(\vec{x}_i - \vec{x}_j\right) / |\vec{x}_i - \vec{x}_j|^2$ // Cohesion, alignment and separation
9:      if $r < c_4^i$ then
10:       $\vec{a} \leftarrow \vec{a} + (r_{\pm 5},\ r_{\pm 5})$ // Random perturbation
11:     end if
12:   end if
13:   $\vec{v}'_i \leftarrow \vec{v}_i + \vec{a}$
14:   $\vec{v}'_i \leftarrow \min(V_m^i / |\vec{v}_i|,\ 1) \cdot \vec{v}'_i$ // Limiting overspeed
15:   $\vec{v}'_i \leftarrow c_5^i (V_n^i / |\vec{v}_i| \cdot \vec{v}_i) + (1 - c_5^i)\vec{v}'_i$ // Self-propulsion
16: end for
17: for all $i \in$ agents do
18:   $\vec{v}_i \leftarrow \vec{v}'_i$ // Update velocity
19:   $\vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i$ // Update position
20: end for
```

3. Methods

3.1. Parameter Optimization

We used a genetic algorithm, known as one of the most widely used parameter optimization techniques, in order to find recipes from which interesting and life-like patterns or behaviors evolutionarily. Specifically, a real-coded genetic algorithm (RCGA) [20] was used because the parameters under optimization take continuous values. The flow of optimization is depicted in Figure 4a. For crossover operation, the blend crossover (BLX-α) method, which is commonly used in RCGA was applied. Figure 4b depicts the concept of BLX-α method. The figure is for the case of having only two parameters to be optimized, for simplicity. Actually, however, the dimension is the same as the number of parameters of the problem, which is 24 in this paper. The parameters of child individual are randomly generated within the domain, indicated with a rectangle surrounded by chain lines, determined from the domain the parameter values of the parent individuals A and B. α is the coefficient for the extent of domain expansion. No mutation was applied since BLX-α itself can control randomness. The magnification rate α is an important parameter to control the optimization. So we have conducted parameter search after tuning this parameter carefully for each fitness function.

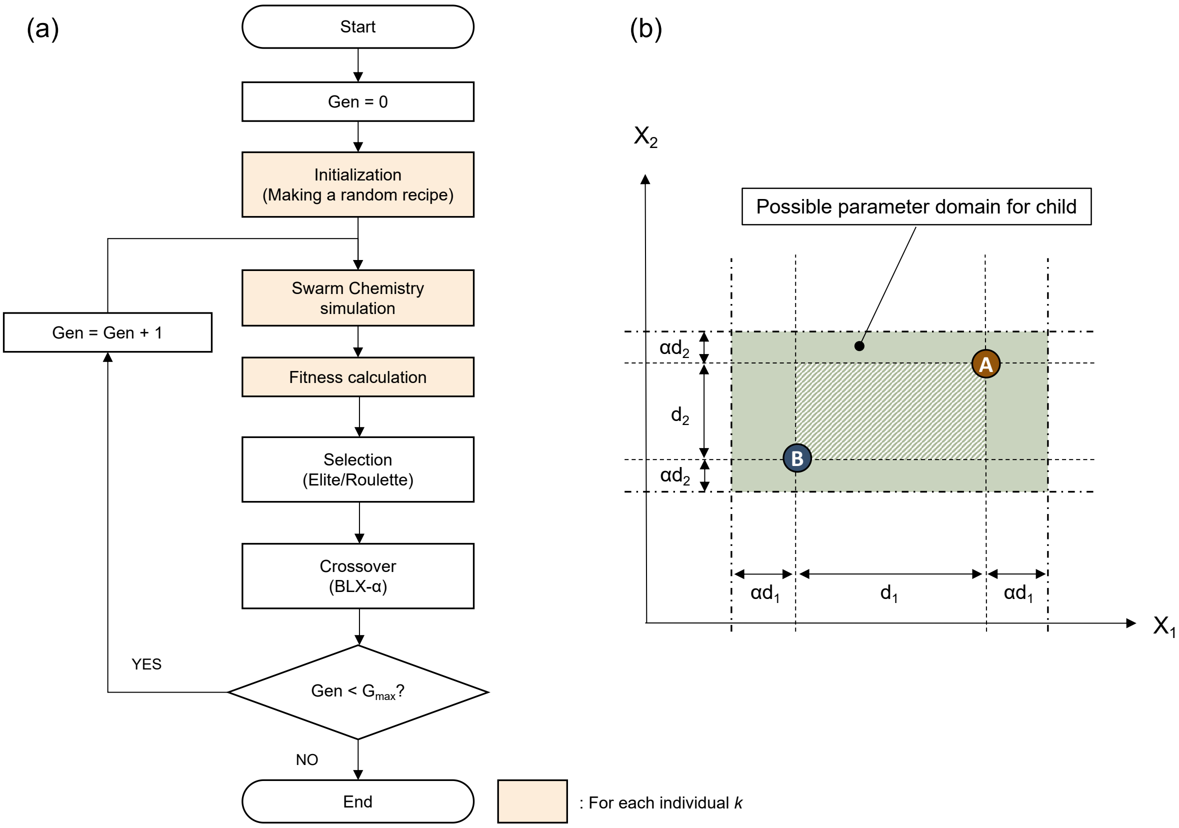


Figure 4. (**a**) The optimization flow with genetic algorithm (GA); (**b**) The conceptual drawing of blend crossover (BLX-α) method.

3.2. Optimization Flow

In each GA iteration (g), each individual (k) has a recipe $\mathcal{R}(g,k)$ input to its Swarm Chemistry simulation. The initial recipes of the individuals are randomly given.

At the beginning of a simulation, agents are placed at random positions within $[0:300] \times [0:300]$ [pixel] on an infinite continuous plane and then move according to the rules shown in Algorithm 1. Thus the system is time-evolved. The simulation is conducted in discrete time steps and finally outputs the history of agent positions, $\vec{x}_i(t) = [x_i(t), y_i(t)]$. Simulation is done for 1000 time steps then the fitness is calculated using the history for the final 500 time steps.

Based on fitness calculated for each individual, blending crossover is applied. In selecting parents, r_E [%] of the highest scoring individuals are chosen and preserved. For the rest of seats, random individuals are chosen by roulette selection, i.e., in proportion to the fitness values.

For simplicity, we set the total number of individuals (n_I) was 100, with the fraction of elite individuals $r_E = 0.2$; $n_E = r_E n_I$. Also, we fixed the number of agent types M to 3 and the number of agents of each type N_j to $300/M = 300/3 = 100, \forall j$, respectively. Overall, we had therefore 24 (8 × 3) parameters to be optimized. The methods of designing effective functions are not limited to the heterogeneous swarms; they can also be applied to the swarms of single type as well. In this case, the model is equivalent to Boids model, whose basic behaviors have already been studied. In Swarm Chemistry, though there are some exceptional cases like "Blobs" shown in No. 2 of Figure 2, the possibility of emergence of "interesting" patterns basically seems to be limited with a single type, compared to the swarms of heterogeneous agents. The interestingness of the emerging behavior depends on the number of types in general. If the number of types is 1, the diversity of the emerging behavior will be limited. On the other hand, for example, it is thought that if the type number is too large, the minimum structure or order does not emerge, as is apparent from the case that the type is different for each particle. In this study, as it is at the primary stage, we have decided to set it to 3. In addition, if the types with different size interact, there is a possibility that some different roles emerge according to the size, which are different from the case where all the types have the same number of agents. We believe it is a promising direction for the future.

4. Measures to Quantify the Swarm Dynamics

4.1. What is an Interesting Pattern?

Before defining the measures, let us discuss what a visually interesting pattern is. It is a premise that providing a strict definition of interestingness is principally impossible since interestingness can be diverse: it can vary from person to person. On the other hand, we believe that there should be certain universal patterns which look interesting to everyone. Some possible conditions, which we have empirically found, in which a swarm looks at least somewhat interesting include:

- A certain regular shape or motion pattern is self-organized
- An organism-like pattern is self-organized
- The swarm pattern is deformed as time evolves
- The role of a type alters as time evolves
- Fission and fusion continuously occur

Of course, there might be some other features to let a swarm look interesting.

On the other hand, it is relatively easy to say what patterns are NOT interesting. Here we show two typical patterns that are uninteresting. One possibility is that it is a necessary condition that the swarm pattern continues to change in time, not falling into a fixed pattern. We consider two typical extreme swarm patterns, which may be uninteresting, illustrated by Figure 5. The first pattern (a) is a simple aggregation, where the agents only gather into a sphere-like structure without changing its shape. Such a motion can be obtained from a recipe, for example:

$$100 \times (244.19, 4.19, 8.74, 0.63, 0.35, 54.69, 0.08, 0.66) \tag{4}$$

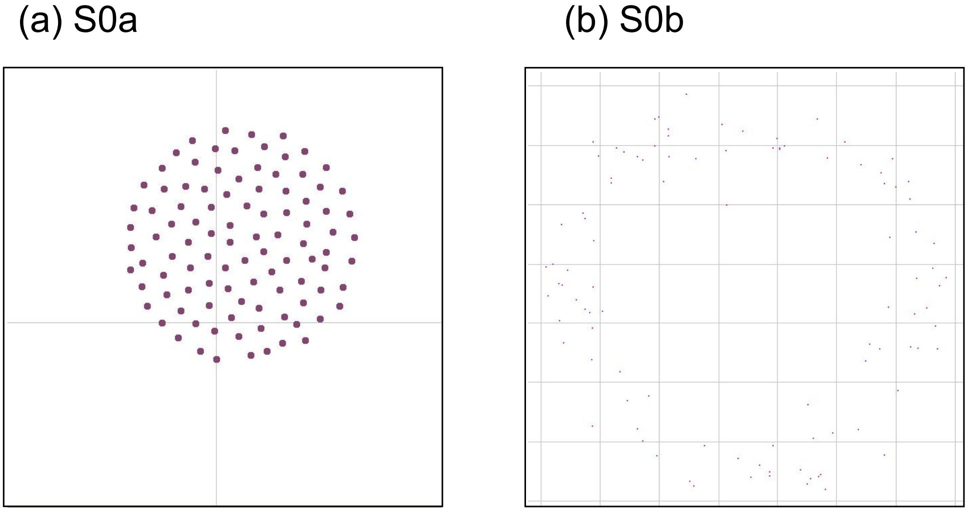


Figure 5. Two extreme swarm patterns we consider as example swarms which seem uninteresting. (**a**) The first pattern "S0a" is a simple aggregating pattern, where the agents only gather into a sphere-like structure without changing its shape. (**b**) Another pattern "S0b" is a simply segregating pattern, where the agents just wander randomly and go far from each other without interacting with any others.

In this paper we call it "S0a". Another pattern is a simply segregating pattern (We use the words "segregation" or "dissipation" to avoid confusion with "separation" in context of original rule of Boids.), where the agents just wander randomly and go far from each other without interacting with any others. Such a motion can be obtained from a recipe, for example:

$$100 \times (9.05, 10.09, 37.8, 0.98, 0.24, 79.97, 0.22, 0.3) \tag{5}$$

In this paper we call it "S0b". These patterns are also included shown later in Table 3.

4.2. Importance of Asymmetry

In biology, for decades (left-right) morphological asymmetries have evoked curiosity and wonder, which is one of those exceedingly rare characteristics of animals that has evolved independently many times [21,22].

In physics, the symmetry principle proposed by Pierre Curie has become the object of renewed philosophical discussion in connection with the growing interest in the role of symmetry and symmetry breaking in recent decades [23], stating that "When certain causes produce certain effects, the elements of asymmetry of the causes must be found in the effects produced" [24].

Although the meanings of asymmetry in both fields are significantly different, inspired by these recent discussions, we assume that asymmetry, which might be in the causes or effects, plays an important role in creating interesting swarm patterns, as a working hypothesis.

4.3. Fitness Function Candidates

Upon the above discussion, we define seven fitness function (FF) candidates from the aspect of asymmetry of some kinds. Table 2 summarizes the features of the fitness functions. FF-A and FF-B treat asymmetries in macroscopic positions. FF-C to FF-F are based on asymmetries related to microscopic interactions. While these six FFs measure explicit asymmetry, the final one, FF-G, measures the system asymmetry implicitly by using chaoticity index. In this context, "implicit" means measuring directly the appeared (resultant) asymmetry. On the other hand, "explicit" means measuring asymmetry by computing the consequent complexity arising from inherent asymmetry of system.

Another classification can also be made according to dynamicity. The functions are classified into either static or dynamic. Here, "static" means that the FF measures the instance arrangement or state whereas "dynamic" means that the FF focuses on the change of arrangement or state in time. From this point of view, FF-A to FF-C are regarded static and the remainders dynamic, respectively.

The detailed description for each function will be provided in more detail in the following subsections.

Table 2. Fitness functions (FF) proposed in this paper.

FF	Meaning	Explicity of Aymmetry	Asymmetry Type	Dynamicity
A	Typewise centroid deviation	Explicit	Positional	Static
B	Variation of pairwise distance			
C	In-/out-degree difference		Topological	
D	Derivative of in-/out-degree difference			Dynamic
E	Sum of derivatives of in-/out-degrees			
F	Fluctuation of in-/out-degree difference			
G	Max. Lyapunov exponent $\lambda_{\max}$	Implicit		

4.3.1. Fitness Function A

One straightforward way to characterize the swarm structure can be the geometrical asymmetry of a swarm. We have defined a fitness function which represents the positional deviation of the centroid of each type from the global centroid. This measure is computed from the snapshots of the swarm.

In this sense, it is a metric of the static structure of swarm. In a mathematical form, this fitness function could be written as:

$$\phi = \frac{1}{M}\sum_{t=1}^{T}\sum_{j=1}^{M}\left\|\vec{x}_{jC} - \vec{x}_{GC}\right\| \tag{6}$$

$$\vec{x}_{jC} = (x_{jC},\ y_{jC}) \tag{7}$$

$$\vec{x}_{GC} = (x_{GC},\ y_{GC})\,, \tag{8}$$

where M is the number of types, $\vec{x}_{jC}$ is the centroid coordinate of j-th type, and $\vec{x}_{GC}$ is the centroid coordinate of all agents. T denotes the whole duration of the simulation. The centroid coordinate of j-th type equals to the $(x,\ y)$ positions averaged over the set of agents belonging to that type j. Similarly, the global centroid coordinate equals to the $(x,\ y)$ positions averaged over all the agents. When the emerged swarm has a unique shape like ones shown in Figure 2, there are cases where the type-wise centroid does not overlap the global centroid while achieving balances between the different types. There is a possibility that maximizing this function leads to an interesting pattern.

4.3.2. Fitness Function B

Another possibility of characterizing the swarm pattern from the asymmetry of static structure can be based on the variation of the pair-wise distance between two agents. We define the fitness function as:

$$\phi = \frac{1}{T}\sum_{t=1}^{T}\sigma\left[D_{ij}(t)\right], \tag{9}$$

where $D_{ij}(t) = \left\|\vec{x}_i(t) - \vec{x}_j(t)\right\|$ is the distance between agent i and agent j at the time step t. This function is the standard deviation σ of the pair-wise distances, measured by the snapshots of the swarm, too. If the structure is a simple ball-like pattern or just dispersing, the pair-wise distances will be uniform. As it is suggested that the distance between two agents at the equilibrium state depends on the ratio of kinetic parameters c_1/c_3 (c_1: cohesion coefficient, c_3: separating coefficient) of these agents [7,10].

4.3.3. Fitness Function C

As one of the most important features of SC, different types of agents are allowed to exist in a swarm, implying the interactions among agents can be asymmetrical. Unlike other swarm system models, the perception ranges R are not always consistent between any pair of agents (i, j), making such a situation possible to occur that the motion of one of them (agent i) is affected by the other (agent j), but not necessarily vice versa. The topological connectivity could be relevant to the emergence of a life-like behavior of agents. From terminology of graph theory, the network topology of interactions can be modeled as a directed graph, whose adjacency matrix is asymmetrical. In preparation, we need to define the in-degree and out-degree functions.

- In-degree (K^-): The number of agents that agent i perceives. In other words, how many agents affect the motion of agent i.
- Out-degree (K^+): The number of agents that perceive agent i. In other words, how many agents are affected by agent i.

A numerical example is illustrated in Figure 6. Consider a swarm heterogeneous of 4 agents. In the case shown in (a), agent 1 is perceived from only agent 3, which means the in-degree of agent 1 is 1. By contrast, agent 1 perceives agent 2, 3 and 4. This means the out-degree of agent 1 is 3. The relationship of perception is visualized in (b). The degrees for the other agents can be computed in the same way. The result is shown in (c).

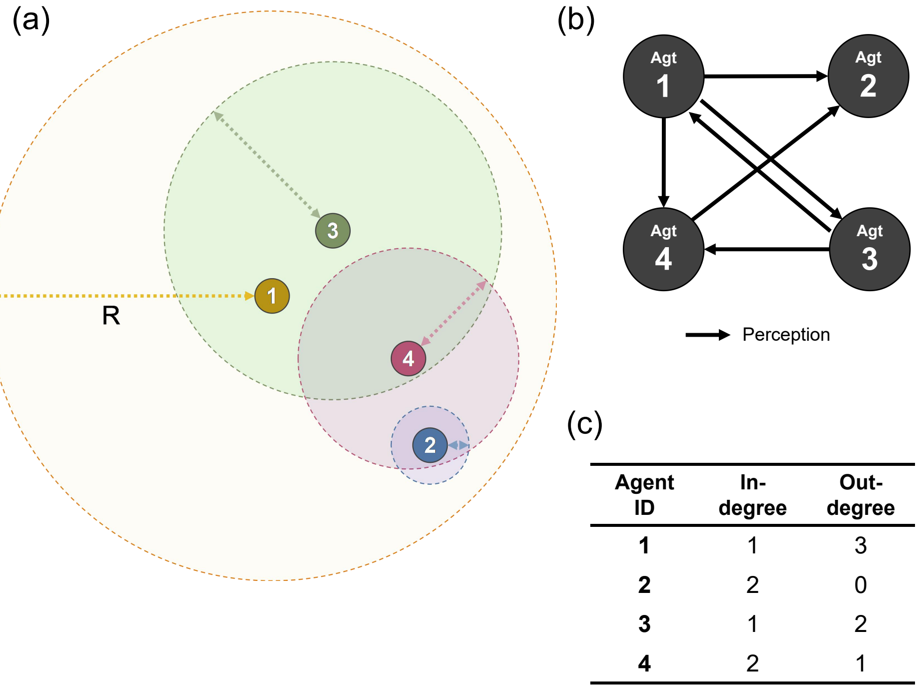
In addition, as the degrees K^+ and K^- are dependent on both time (t) and agent ID (i), they can be formally written as:

$$K^-(t,\ i) \text{ and } K^+(t,\ i), \tag{10}$$

respectively. After every simulation, an adjacency matrix $A = [a_{ij}]$ was computed for each timestep. Let $a_{ij} = 1$ if agent i "perceives" agent j, otherwise $a_{ij} = 1$, and vice versa.

Based on the above preparation, a fitness function could be defined as follows.

$$\begin{aligned} \Delta K(t,i) &= \left|K^+(t,i) - K^-(t,i)\right| & (11) \\ \phi &= \frac{1}{TN^2} \sum_{t=1}^{T} \sum_{i=1}^{N} \Delta K(t,i) & (12) \end{aligned}$$



Agent ID	In-degree	Out-degree
1	1	3
2	2	0
3	1	2
4	2	1

Figure 6. Definition of in- and out-degrees. (**a**) An example swarm consisting of 4 agents; (**b**) Its perception relationship; (**c**) The in- and out- degrees.

4.3.4. Fitness Function D

$$\phi = \frac{1}{TN^2} \sum_{t=2}^{T} \sum_{i=1}^{N} |\Delta K(t,i) - \Delta K(t-1,i)| \tag{13}$$

This function means the time derivative of in/out difference. Using not only ΔK itself, we assumed that it could make it possible to detect the dynamic structural change.

4.3.5. Fitness Function E

We introduce another function similar to FF-D.

$$\begin{aligned}\Delta K(t,i) &= \left|K^{+}(t,i)-K^{-}(t,i)\right| &(14)\\ \phi &= \frac{1}{2TN}\sum_{t=2}^{T}\sum_{i=1}^{N}\left[\left|K^{+}(t,i)-K^{+}(t-1,i)\right|+\left|K^{-}(t,i)-K^{-}(t-1,i)\right|\right] &(15)\end{aligned}$$

As will be shown later in the result section, the optimization result from FF-D looked interesting. Therefore we came to a hypothesis that a derivative operation could help capture a pulsating or alternating motion. We investigate whether it is still possible to get a similar kind of motion as well by this fitness function.

What this function differs from FF-D is that the derivatives are taken for K^{+} and K^{-}, respectively.

4.3.6. Fitness Function F

Another approach based on the topological connectivity could be defined as:

$$\begin{aligned}\Delta K(t,i) &= \left|K^{+}(t,i)-K^{-}(t,i)\right| &(16)\\ \phi &= \frac{1}{N}\sum_{i=1}^{N}\left[\max_{t}\left(\Delta K(t,i)\right)-\min_{t}\left(\Delta K(t,i)\right)\right]. &(17)\end{aligned}$$

This function measures the average of the amplitude of in/out difference over all time range. By this function we expected to get more gradual structural formation and deformation, while the previous function in Equation (13) measures the temporal variation between two successive time steps, leading to a snappy motion, as described in the result section.

4.3.7. Fitness Function G

The behaviors of swarms in SC can be seen as a kind of nonlinear dynamic system. We assume inherent asymmetry might exist in the system dynamics. Therefore, this concept was woven into a form of fitness function. The FF is related to the instability or the unpredictiveness.

Let the state of system be denoted using a vector $\vec{x}$, which represents the agent positions in the two-dimension:

$$\vec{x}_i = \left[x_{1i}, x_{2i}\right]^{T}. \tag{18}$$

Hence, the system state vector is defined as:

$$\vec{X} = \left[\vec{x}_1^{T}\ \vec{x}_2^{T} \ldots \vec{x}_N^{T}\right]^{T}. \tag{19}$$

Namely, $\vec{X} \in R^{m}$, $m = 2N$, N is the number of agents. As $\vec{X}$ time-evolves, it can be written as $\vec{X}(t)$.

A Lyapunov exponent λ is often used to determine if a dynamical system is chaotic. Lyapunov exponent is defined as:

$$\lambda = \frac{1}{T}\sum_{t=1}^{T}\log_{10}\left|\frac{\Delta X(t)}{\Delta X(t-1)}\right| \tag{20}$$

A Lyapunov exponent indicates the slope λ when the expansion of system state locus is approximated by an exponential function $e^{\lambda t}$. Thus the stability of the system can be determined by the following expressions.

$$\begin{aligned} \lambda &> 0 \Rightarrow \text{unstable (chaotic)} \quad (21)\\ \lambda &= 0 \Rightarrow \text{critical} \quad (22)\\ \lambda &< 0 \Rightarrow \text{stable} \quad (23) \end{aligned}$$

As Lyapunov exponent is computed for each dimension of the state vector $\vec{X}$, m Lyapunov exponents $\{\lambda_1,\ \lambda_2, \ldots, \lambda_m\}$ (Lyapunov spectrum) are obtained for m dimensions. We used the maximum λ_{max} as a measuring index because it determines the convergence of solution locus along time evolution of system.

From this, the fitness function is

$$\phi = c\left(e^{\lambda_{max}} - b\right) \quad (24)$$

Here, an exponential function is used so that the fitness function only takes positive values. In addition, the constants c and b were introduced as a magnification coefficient and an offset constant, respectively, and set $c = 1000$ and $b = 1$ empirically because the expression $e^{\lambda_{max}}$ itself gives values around 1 and the difference is small (the order of 10^{-3}). We have introduced these correcting constants because if these parameters were not involved (i.e., $c = 1$ and $b = 0$) in the beginning, the difference in fitness values between the individuals was too small to be discriminated in optimization process. If the magnification factor c is larger, it may be easier to discriminate the fitness difference between individuals. On the other hand, when it is too large, the probability that the individuals with low fitness cannot be selected. So we have chosen a moderate value whose order is similar to that of the averaged fitness values from the case these parameters were not involved. The constant b has been introduced to remove the offset before magnifying by c. This value should be positive but not exceed the averaged offset value, from the case these parameters were not involved, in order to keep fitness values greater than zero.

4.4. Numerical Examples

Using the proposed fitness functions, the fitness values have been computed for already found interesting patterns shown in Table 2. The results are plotted in Figure 7. We can see from this figure that the FFs can be classified roughly into two groups. For the first group (FFs-A, B and G), the variation is small and every swarm takes similar fitness scores. For the other group (FFs-C, D, E and F), only some swarms take a high fitness score. Next, looking at the swarm, some swarms take a specifically high score for the latter group of FF. For example, the swarm No. 10 takes a preeminently high score in FFs-D and E. It is thought that the reason is that the derivative function could capture intensively alternating motion. Overall, we have found that each FF has a diverse individuality and could produce quite different distributions of fitness scores.

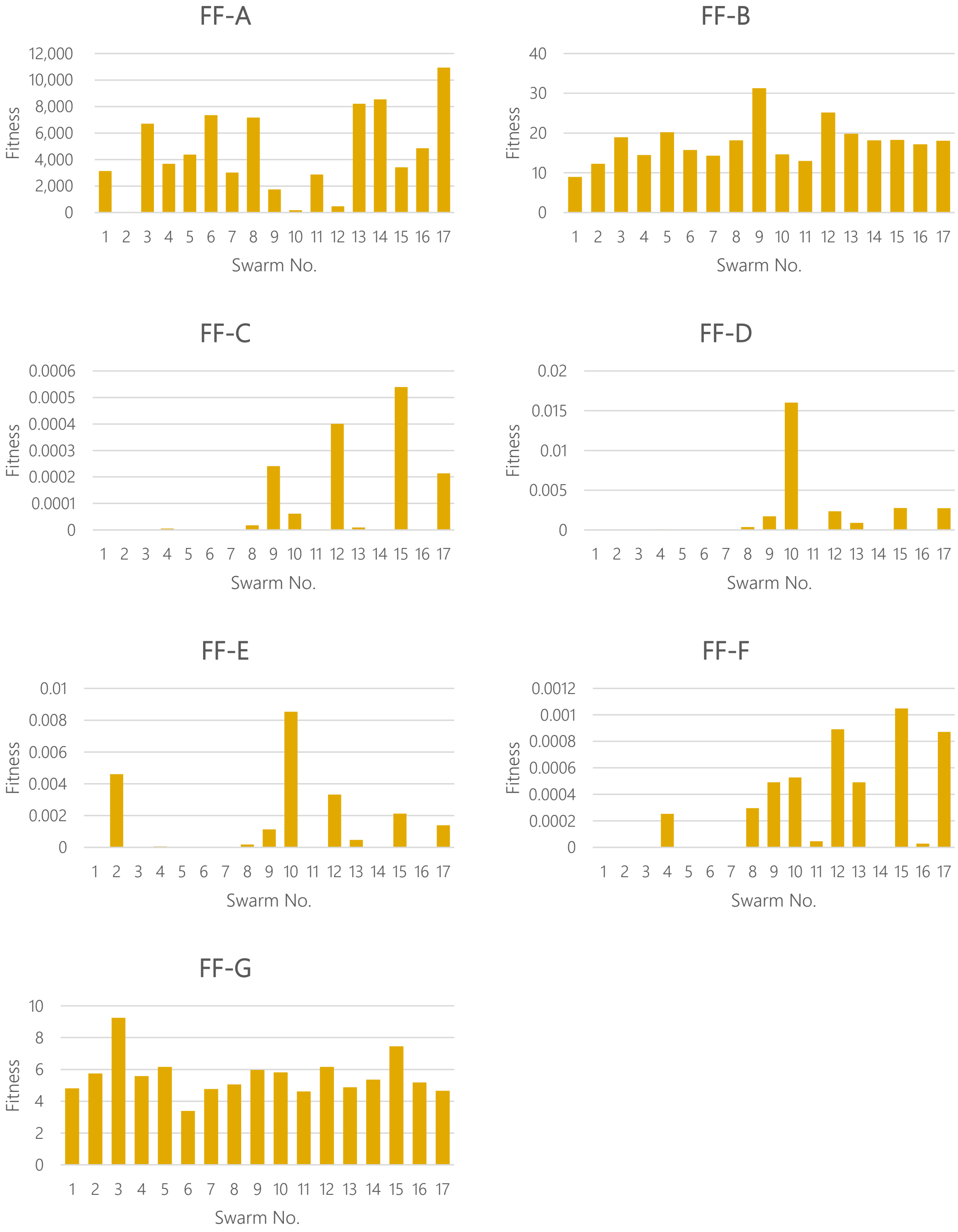
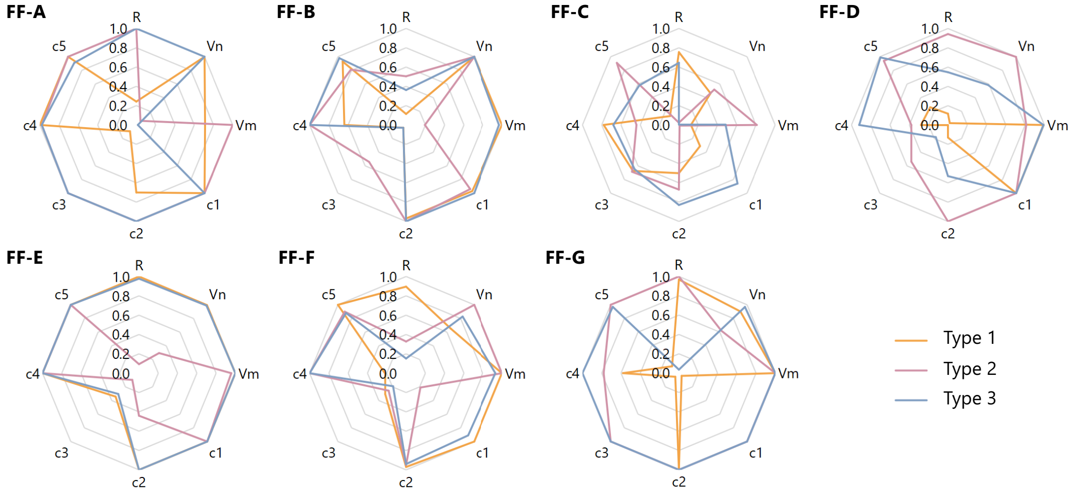


Figure 7. The fitness values computed for already found patterns shown in Table 2, according to the proposed fitness functions.

5. Results

5.1. Summary of Optimization Results and Obtained Recipes

We have carried out parameter searches by means of the GA. The evolutionary change in fitness for fitness functions FF-A to FF-G are shown in Figure A1a–g, respectively. The best recipes $\mathcal{R}$ gained through optimization for each FF are summarized in Figure 8. Note that the KPs in recipes here are normalized with the upper limit of each KP. The result will be discussed later in the next section.



Statistics	N_j	R	V_n	V_m	c_1	c_2	c_3	c_4	c_5
	# of agents (fixed)	Radius of local perception range	Normal speed	Maximum speed	Cohesive force	Aligning force	Separating force	Probability of random steering	Tendency of self-propulsion
Min	0.333	0.023	0.005	0.016	0.010	0.130	0.002	0.220	0.100
Max	0.333	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Mean	0.333	0.557	0.710	0.809	0.821	0.840	0.428	0.793	0.823
Standard Deviation	0.000	0.385	0.357	0.320	0.344	0.249	0.356	0.268	0.293

Figure 8. Optimization results: the recipes gained using the proposed fitness functions. Each KP is normalized with its upper limit value.

5.2. Evolved Swarm Behaviors

This subsection presents the evolved swarming behaviors obtained through optimization. The snapshots for each fitness function are also shown in Figures A2–A8, respectively. The different types of agents are shown in different colors. The semi-transparent dots behind the agents also show their past positions at the recent 60 time steps, allowing us to understand the direction of movement of each agent. Animated versions of swarm motions are also available as Supplementary Materials, Videos S2–S8.

5.2.1. Fitness Function A

Type 1 agents (colored in green) quickly got apart of the other types and no interesting pattern was formed. In this sense, it seemed this fitness function itself was not well defined.

Although, we could find an interesting motion appearing from this recipe. Beginning simulations with several different initial conditions (i.e., agent positions), it was found that two distinctive patterns appear from this recipe (see Figure 9): while the aggregation size of yellow agents is small, they are captured within the "gel-structure" composed of the grey agents. But once the yellow aggregation of agents gathers in number, the aggregation gets out of the gel and goes away. Such a behavior might happen near a critical condition. Whether the yellow agents could escape from the gel or not depends on the sensitive balance of forces applied to the agents. The number of aggregations was occasionally two or three.

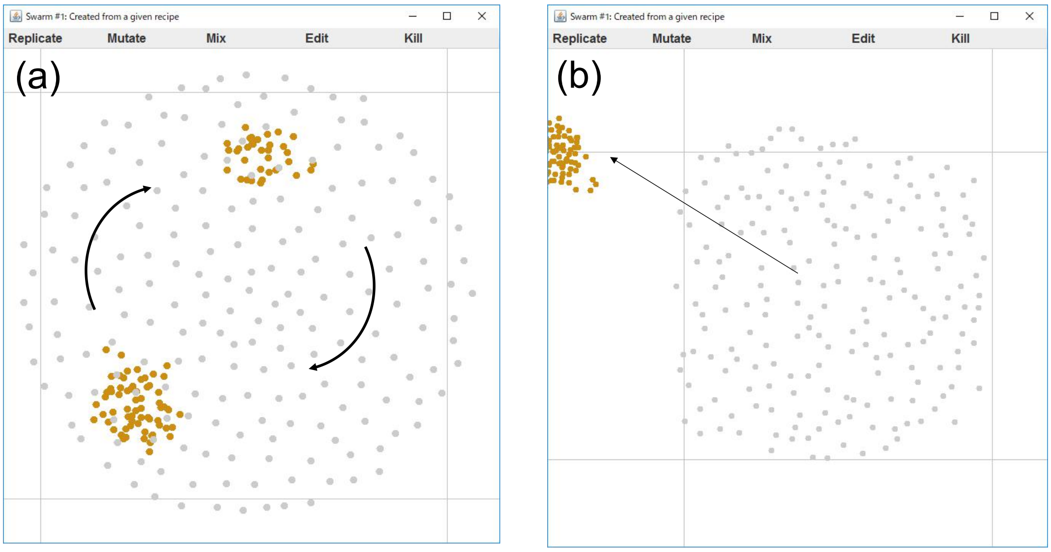

Swarm #1: Created from a given recipe
Replicate
Mutate
Mix
Edit
Kill
(a)
Swarm #1: Created from a given recipe
Replicate
Mutate
Mix
Edit
Kill
(b)

Figure 9. Different motions observed in the FF-A. (**a**) While the aggregation size of yellow agents is small, they are captured within the "gel-structure" of the grey agents; (**b**) As soon as the yellow aggregation of agents gathers in number, the aggregation gets out of the gel and goes away.

5.2.2. Fitness Function B

One type (Type 1) of agents quickly go far and segregate from others and the interaction connection is soon broken off. Here, Type 1 has a small perception range R and a relatively high speed. This will lead the Type 1 agents to quick segregation.

5.2.3. Fitness Function C

This FF resulted in a dispersing pattern, a little different from the results in FFs-A and B. In those FFs, one type get out quickly in aggregation, while in this FF, one type (Type 2) dispersed outwards. Type 2 has a small perception range R and a relatively high speed.

5.2.4. Fitness Function D

A shell-like structure and a snappy motion like convulsion, blinking or pulsation can be observed. This kind of motion seems to be in the same category of "pulsating eye", shown in Figure 2 (No. 10). Figure 10 shows its representative dynamics extracted , with the snapshots over 4 successive timesteps: (a) is the snapshot at a timestep t, (b) at $t+1$, (c) at $t+2$, and (d) at $t+3$, respectively. We can see that the radii of the outer and inner shells increase and decrease in odd and even steps alternately. This is due to the attraction and repulsion forces (i.e., cohesion and separation, respectively). This can be seen as that contraction and relaxation are repeated, as we can also see by looking at the shadows of the Type 3 agents (colored in red) in e.g., Figure A5e–i. This swarm successfully sustained its self-organized structure without segregation. Even though the perception range of Type 1 is relatively small, the speed is also small. So the Type 1 can keep staying in the structure.

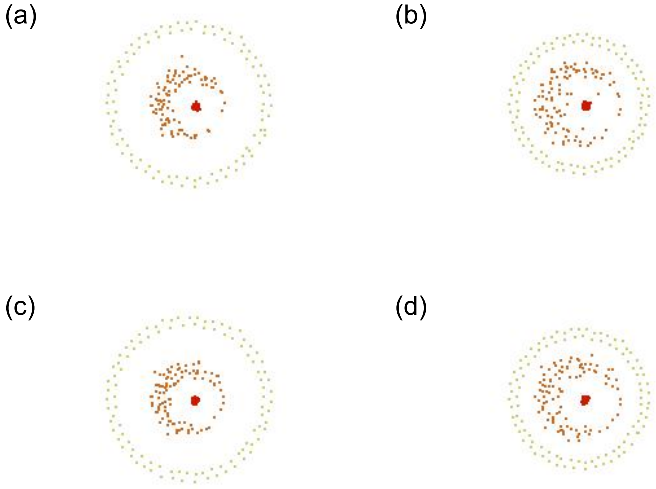


Figure 10. Representative dynamics of the swarm obtained from FF-D. (**a**) The snapshot at a timestep t; (**b**) at $t+1$; (**c**) at $t+2$; and (**d**) at $t+3$, respectively. We can observe a blinking motion: the radii of outer and inner shells increase and decrease in odd and even steps alternately. This is due to the attraction and repulsion forces (i.e., cohesion and separation, respectively) are acting.

5.2.5. Fitness Function E

Figure 11 shows its representative dynamics. This motion looks like a pulsation, blinking or hiccup, like the one appearing in the previous FF-D. The same arrangements appear every 2 time steps alternately, and the agents of yellow type go back and forth. Again, Type 2, which has a small perception range and a low speed, is able to stay in a structure. And Types 1 and 3 are almost the same type.

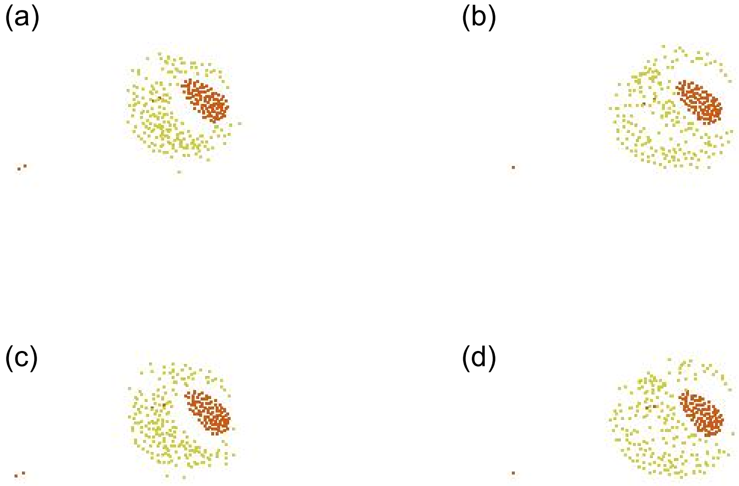


Figure 11. Representative dynamics of the swarm obtained from FF-E. (**a**) The snapshot at a timestep t; (**b**) at t + 1; (**c**) at t + 2; and (**d**) at t + 3. Similar to the result from FF-D, the same arrangements appear every 2 time steps alternately. The agents of yellow type go back and forth.

5.2.6. Fitness Function F

Figure A7 illustrates the agent trajectories and its characteristic dynamics is depicted in Figure 12. This recipe led us to a dynamics exhibiting a kind of motion like oviposition. The shape continues to deform during the transient from an egg-like structure to eventually fission into several aggregations.

It looks interesting also because the three types temporarily composed a three-layer structure and experienced a two-stage segregation. In the first segregation was that Type 2 agents (blue) created an elastic tube-like structure and the other types (Types 1 and 3) created an egg-like structure flowing down the tube. The subsequent segregation was that of Type 1 (red) agents as an egg from Type 3 agents (green) as a tube. Finally, each type grew into isolated eggs. This FF led to a life-like or lively change of shape: some types experienced both roles as a tube and as an egg in the transient history. While observing the motion, we can also feel something viscous during the fission process above. This viscosity is thought to be due to some complex forces around the interface between the types.

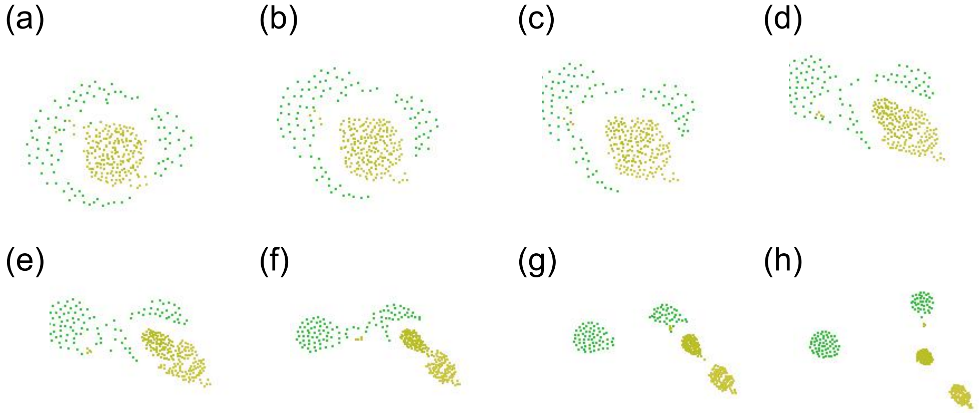


Figure 12. Representative dynamics of the swarm obtained from FF-F. (**a**) The snapshot at a timestep t; (**b**) at t + 10; (**c**) at t + 15; (**d**) at t + 35; (**e**) at t + 40; (**f**) at t + 60; (**g**) at t + 75; and (**h**) at t + 105. The shape continues to deform from an egg-like structure to eventually fission into several aggregations.

5.2.7. Fitness Function G

This recipe also results in a swarm like FF-C, undergoing segregation and not interesting pattern can be formed. The main reason for the segregation is the perception range R of Type 3 which is too small to keep the connection, because if this value 10.82 is replaced by 200, then a structure is retained.

6. Discussion

6.1. Overview of Results

According to their dynamical motions, we have roughly classified the patterns including already found ones (S1 to S17) shown in Figure 2 and uninteresting ones (S0a and S0b), into several categories. This classification is shown in Table 3. The results from the seven fitness functions proposed in this paper are also in this table.

Defining a good measure was found to be not easy: there are not few cases where even uninteresting patterns could get a high fitness score. A swarm tends to result in a segregating pattern, which is frequently observed but uninteresting, when one or more types have a too small perception range and a too large speed. Therefore, introducing some terms of limiting into the fitness functions may be effective to prevent an easy breaking of the structure.

The recipes gained through the optimization are shown in Figure 8. We can see from this table that:

1. Some types have a small perception range R and a large normal speed V_n (e.g., Type 1 in FF-A, Type 1 in FF-B, Type 2 in FF-C, Types 2 and 3 in FF-F and Type 3 in FF-G). On the contrary, some other types have a large R and a small V_n (e.g., Type 1 in FF-C, Types 1 and 3 in FF-D, Type 1 in FF-F, Types 1 and 2 in FF-G).
2. Similarly, some types have a small cohesion coefficient c_1 and a large separation coefficient c_3 (e.g., Types 1 and 2 in FF-C), and vice versa.
3. In FF-E, Types 1 and 3 share the almost identical KPs.

Especially, as can be seen in items 1 and 2 above, the relationships between the perception range R and the normal speed V_n as well as between the cohesion coefficient c_1 and the separation coefficient c_3 seem important to control the dispersion behaviors.

From the above results together with the dynamics obtained, we discuss the relationship between the recipe (KPs) and the resultant swarm patterns. Dispersive motions like FFs-A, B, C and G are observed when: the swarm has one or more types whose perception range R is small whereas normal speed V_n is large; and/or separation coefficient c_3 is large whereas cohesion coefficient c_1 is small. Therefore, it seems better to keep c_1 large and c_3 small, as well as to keep R large and V_n small, in order to create a linked formation preventing from a dispersion.

Figure 8 also summarizes the statistical tendency, telling us that:

1. As the standard deviations are small and the mean values are relatively high (approximately 0.8) in c_2 (aligning coefficient), c_4 (randomness) and c_5 (tendency of self-propulsion), implying that these parameters are almost common and less sensitive to the definition of fitness functions than the other ones.
2. On the other hand, diversity is large for R (perception range), V_n (normal speed), V_m (maximum speed), c_1 (cohesive coefficient) and c_3 (separating coefficient), implying that these parameters are sensitive to the definition of fitness functions.
3. In addition, the mean values of c_1 tends to be large and that of c_3 is small.

Table 3. Classification of typical motions observed in Swarm Chemistry (SC).

Category	Kind of Motion	Swarm No.
G1	Aggregating	S0a
G2	Segregating	S0b, **FF-A**, **FF-B**, **FF-C**, **FF-G**
G3	Reciprocating	S1, S8, S13
G4	Rotating	S5, S11, S17
G5	Tracting	S3, S4, S6, S7, S14
G6	Pulsating	S10, **FF-D**, **FF-E**
G7	Clustered	S2, S9, S12
G8	Cell-like	S15, S16
G9	Viscous interfacing	**FF-F**

6.2. Analysis of Especially Interesting Patterns

In this subsection we would like to get into a little more detailed discussion especially on the most interesting swarm patterns shown in the previous subsection, analyzing how such a motion could be obtained. From the classification above, let us pick up the result from FF-D, FF-E and FF-F. Notable dynamical patterns have been yielded from FFs-D, E and F. FFs-D and E have delivered some motion like a pulsation. Their corresponding swarm patterns keep alternating motion and its formation without dissipating. FF-F has delivered a gradually changing structure. It is also interesting that the type could experience different roles in the same transient history. The other FFs resulted in simply segregating motions, where agents tend to go far and the interactions between them are broken off.

We can see that the asymmetry of information transfer, which is represented by the asymmetrical topological connectivity, is an effective way to characterize the interestingness of heterogeneous swarm patterns. As indicated by FFs-D and E, the usage of derivative function can be seen as it helps detect a pulsating motion. The differential operation seems to facilitate the sustainment of quick transitions between high-degree and low-degree states over two successive time steps. The reason is if the degree monotonically increased or decreased, the fitness value would have been low. This argument may not be limited only to the case of using the degrees, but applicable also to some other measuring indices representing the state of system. The difference in the definition of the fitness functions between FF-D and FF-E is that FF-D uses the time-derivative for the in-/out-degree difference ΔK, whereas FF-E uses the sum of time-derivatives for each of in and out degrees, calculated separately. The optimization results from both FFs have converged into a very similar pattern: a pulsating pattern. However, as shown in the new Figure 5 (fitness functions calculated for Sayama's 17 swarms), a difference to be addressed is that FF-D excludes the swarm of a single type because ΔK is always zero for such swarms, whereas FF-E includes the swarm of a single type.

FF-F could lead to a gradual structural deformation, which looks lively. The contributing factor is supposed to be the long evaluation period which starts at a sufficiently time-elapsed point. If a swarm quickly segregates in the early time steps of the simulation, the min-max difference at the period of evaluation will not be so large. Therefore, it is suggested that the combination of a measuring function and an evaluation period might help detect a slowly deforming swarm pattern.

7. Conclusions

The present study has been motivated by the practical needs to the potential of the emergence of interesting patterns in SC has not been totally revealed and leveraged. It is better to be able to detect measure the interestingness of the static structures of the swarms or the dynamic changes in them, in an objective manner, e.g., by GA.

This paper has studied an effective way of quantifying the interestingness of swarm patterns that emerge in SC and then to detect such interesting patterns automatically. We have defined several quantitative measures, focusing particularly on the asymmetry of some kinds. While swarms tend to result in a segregating pattern, which is not interesting, several notable dynamical patterns have been yielded from the proposed FFs. FFs-D and E have delivered some motion like a pulsation, and FF-F has delivered a gradually changing structure. It is also interesting that the a type could experience different roles in the same transient history.

The contributions of this paper are summarized as follows.

1. We have constructed a framework to search higher-scoring swarms using a genetic algorithm. Experimental results showed the possibility of detecting several characteristic and attractive patterns using the proposed quantitative measures and optimization framework.
2. Discussing the relationship between the fitness functions and the corresponding swarm behaviors or structures gained by them, we have derived some key points of designing a good function to detect interesting patterns.

The results provide some clue to what kind of fitness function should be used to obtain a specific type of structure or motion. Numerical study using seven different fitness functions has given a brief overview and general knowledge about the quantification of interestingness. As a general conclusion on how we can define the interestingness, the geometrical (positional) asymmetry of structures and topological asymmetry of information transfer have possibilities to let interesting swarm patterns emerge. Major findings are summarized as below.

1. A swarm tends to result in a segregating pattern, which is frequently observed but uninteresting, when one or more type has a too small perception range and a too high speed. Therefore, it is better to choose values for a recipe with a large perception range, a speed which does not overwhelm the perception range. Introducing some terms of limiting into the fitness functions may be effective to prevent an easy breaking of structure.
2. Another possibility of avoiding a segregating patterns is making the cohesion force large and the separation force small.
3. The asymmetry of information transfer, which is represented by the topological connectivity (used in FFs-D, E and F), is an effective way to characterize the interestingness of heterogeneous swarm patterns.
4. The usage of derivative function can help detect a pulsating motion (finding from FFs-D and E). The differential operation is believed to be facilitating quick transitions between high-degree and

low-degree states over two successive time steps to be sustained, while preventing from tendency of breaking off. The fitness value would have been low if the degree monotonically increased or decreased. The same discussion might be applied only to the case of using the degrees, but also to some other measuring indices representing the state of system.

5. A gradual deformation, which looks lively or like an organism, can be captured by a function form like FF-F, in combination with a long evaluation period which starts at a sufficiently time-elapsed point, because a swarm with quick segregation occurring in the early time steps will not have so large min-max difference at the period of evaluation and eventually its fitness score will be low.

Recently, Kano and others have presented a minimal model of swarming behavior, inspired from friendship formation in human society [25]. Simulation results showed that emerged patterns can be classified into 6 categories by introducing two macroscopic variables. Comparative study of this and our models would be fruitful.

The proposal may also have applications to the more practical situations. For instance, the pulsating pattern could be applied to the formation control of excavating robots as its motion pattern seems to be able to shatter the rock walls. Swarm robots, with controlling recipes like the ones presented in this paper, could be introduced to such tasks, where every single robot is inexpensive and easily replaceable because the robots have a self-repairing ability. Of course, in order to realize it in the real-world robots, we require further considerations such as the mass (volume) effects, the strength of materials of the body and the delays of operating commands.

Additionally, we have chosen 2D for the first step of the study, since 3D is more computationally intensive task. However, the extension to 3D version should be one of the most important directions because we can expect to observe much more complex swarm behavior including twisting movement.

Though it is still a challenging problem to fully define the interestingness, our results have presented some part of possible answers. The achievement will help the design of swarms that one desires. Of course, the fitness functions or measures appeared in this paper are just a few examples and there might still be other possible ways to find further interesting patterns. They await to be revealed in the future.

Supplementary Materials: To support the readers' visual understanding of the swarm dynamics we have uploaded the animation files online. Video S1: Animation of already reported swarm dynamics listed in Table 2 (https://youtu.be/eLJwa0jz5GQ), Video S2: Swarm dynamics obtained for FF-A (https://youtu.be/QSyZHCI-ffo), Video S3: Swarm dynamics obtained for FF-B (https://youtu.be/3byzpQqtcpU), Video S4: Swarm dynamics obtained for FF-C (https://youtu.be/cCreYJUASoQ), Video S5: Swarm dynamics obtained for FF-D (https://youtu.be/QuGpkG5mI4I), Video S6: Swarm dynamics obtained for FF-E (https://youtu.be/gz5Nk98sRLQ), Video S7: Swarm dynamics obtained for FF-F (https://youtu.be/B888eKL400Q), Video S8: Swarm dynamics obtained for FF-G (https://youtu.be/Dr1RDinc4AU), Video S9: Swarm dynamics of S0a (https://youtu.be/hImZelcStVE), Video S10: Swarm dynamics of S0b (https://youtu.be/02qQYx4cxQ0).

Author Contributions: Naoki Nishikawa, Reiji Suzuki and Takaya Arita conceived and designed the experiments; Naoki Nishikawa performed the experiments, analyzed the data and wrote the paper.

Acknowledgments: This work was supported by MEXT/JSPS Grant-in-Aid for Scientific Research on Innovative Areas #4903 (Evolinguistics), Grant Number JP17H06383.

Conflicts of Interest: The authors declare no conflict of interest.

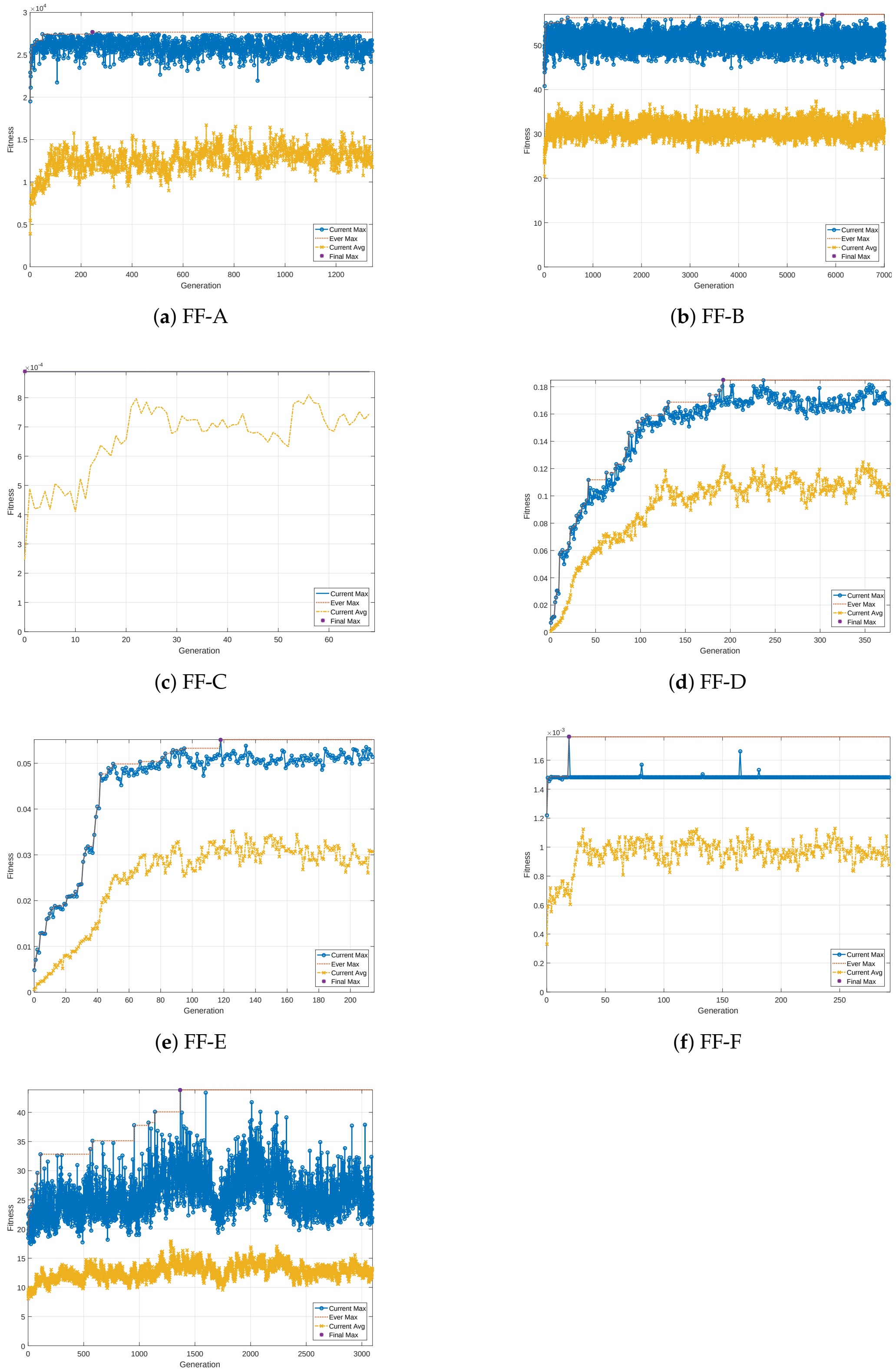
Abbreviations

The following abbreviations appeared in this manuscript:

SC	Swarm Chemistry
ESC	Evolutionary Swarm Chemistry
KP	kinetic parameter
GA	genetic algorithm
FF	fitness function

Appendix A Whole Dynamics for Each Fitness Function

The snapshots for each fitness function are also shown in Figures A2–A8, respectively.



(**a**) FF-A

(**b**) FF-B

(**c**) FF-C

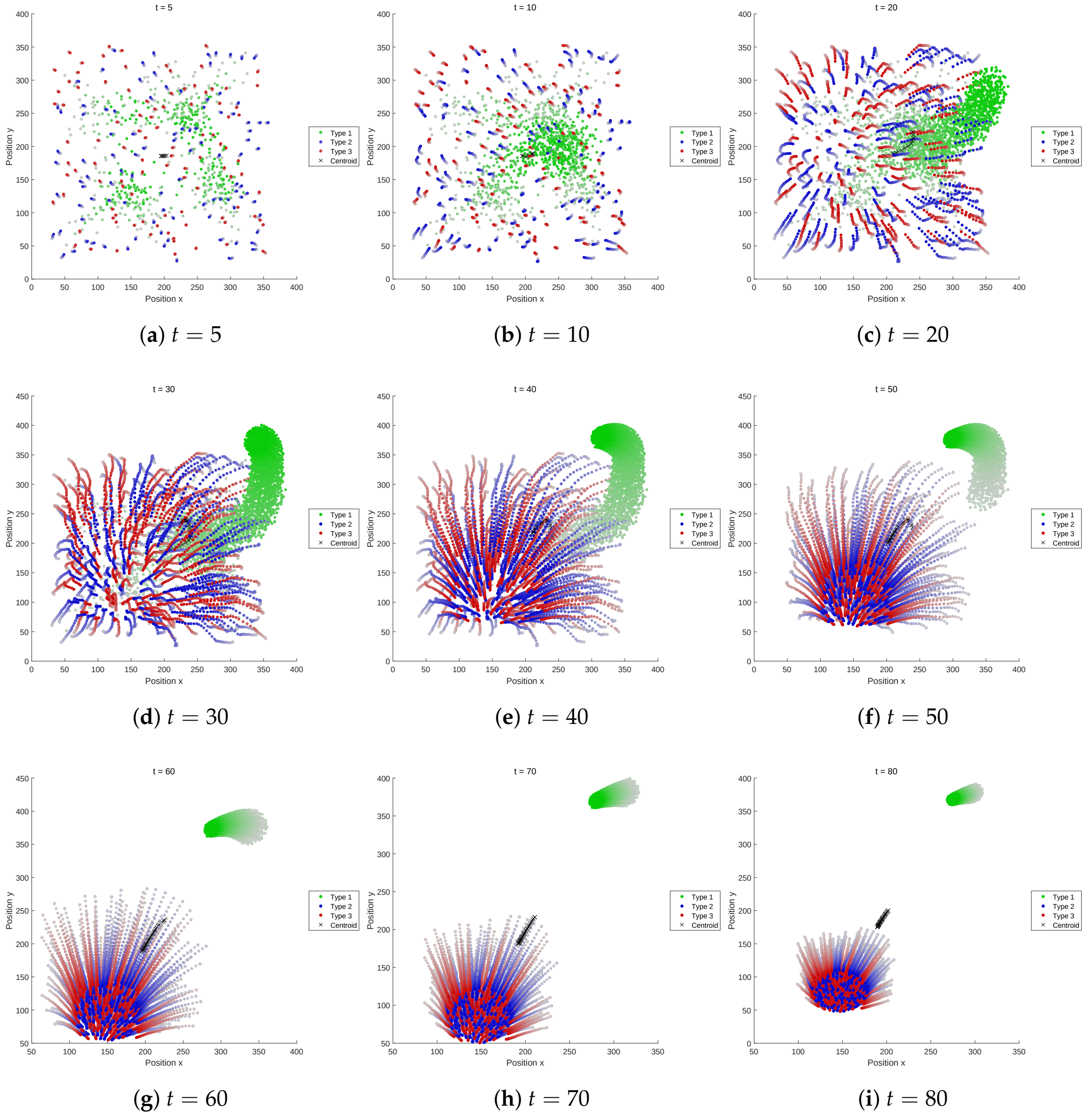
(**d**) FF-D

(**e**) FF-E

(**f**) FF-F

(**g**) FF-G

Figure A1. Evolutionary change in fitness for each fitness function.

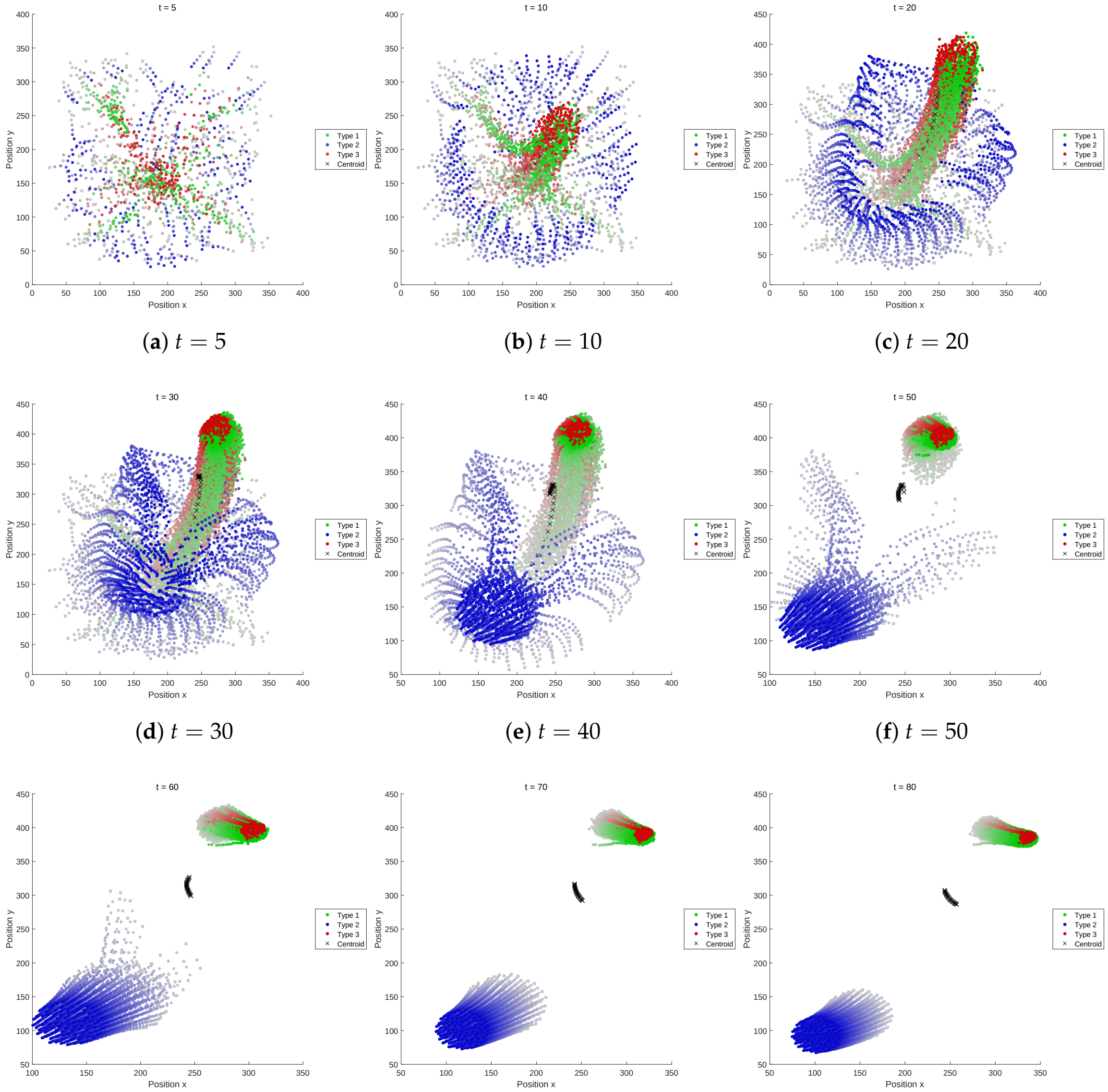


(**a**) $t = 5$ (**b**) $t = 10$ (**c**) $t = 20$

(**d**) $t = 30$ (**e**) $t = 40$ (**f**) $t = 50$

(**g**) $t = 60$ (**h**) $t = 70$ (**i**) $t = 80$

Figure A2. Trajectory of agents (FF-A). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.

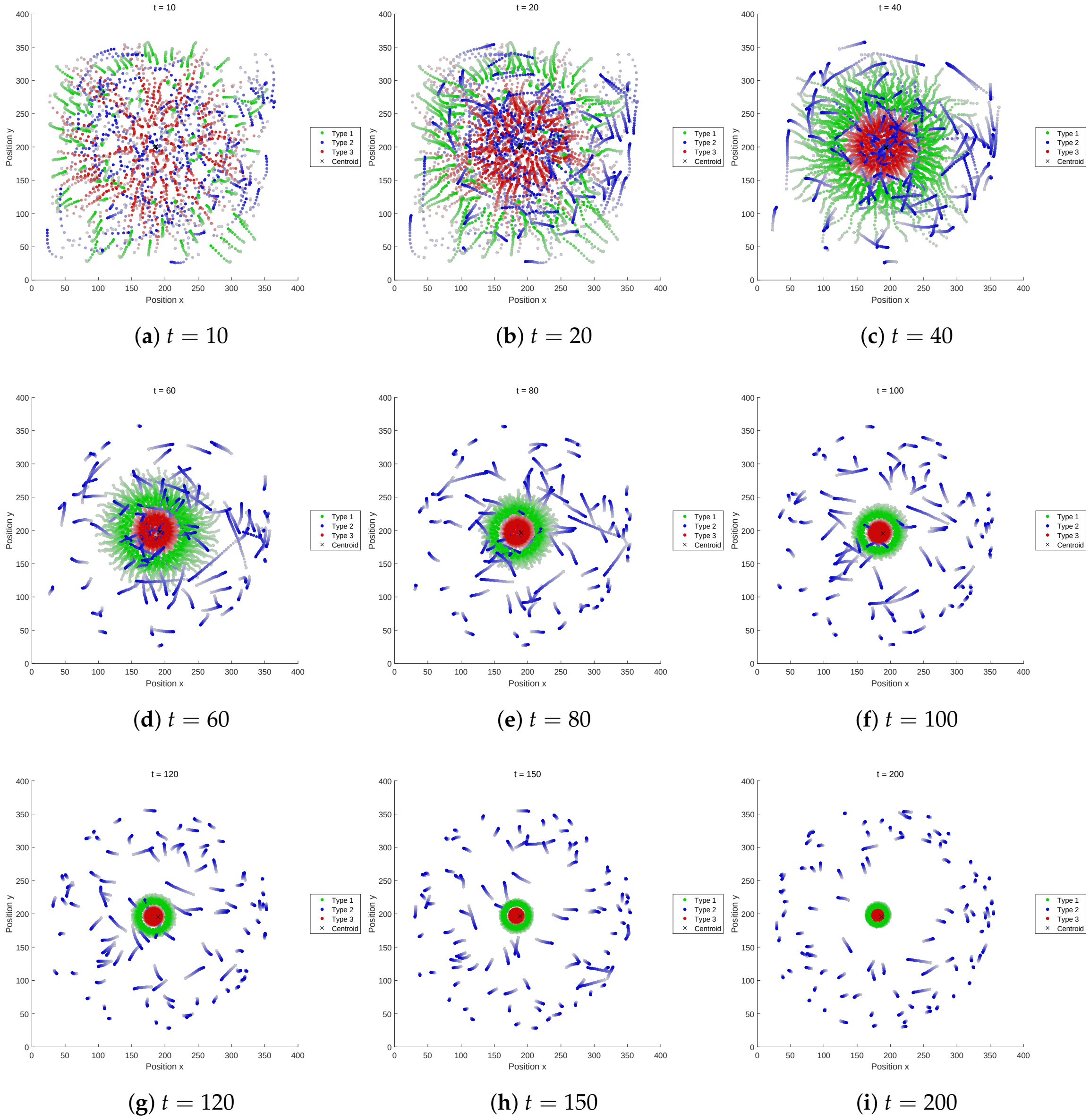


(**a**) $t = 5$ (**b**) $t = 10$ (**c**) $t = 20$

(**d**) $t = 30$ (**e**) $t = 40$ (**f**) $t = 50$

(**g**) $t = 60$ (**h**) $t = 70$ (**i**) $t = 80$

Figure A3. Trajectory of agents (FF-B). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.

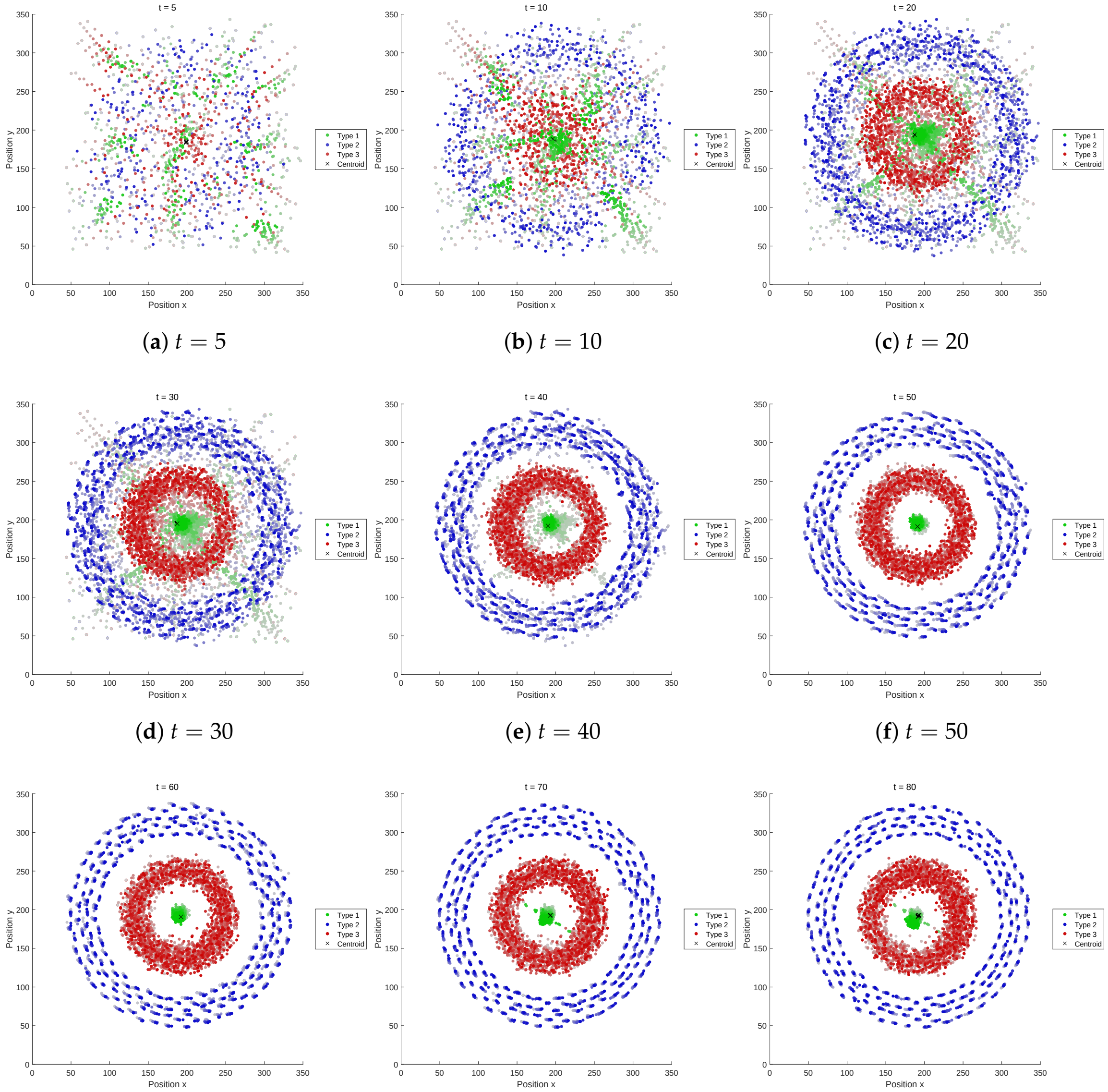


(**a**) $t = 10$ (**b**) $t = 20$ (**c**) $t = 40$

(**d**) $t = 60$ (**e**) $t = 80$ (**f**) $t = 100$

(**g**) $t = 120$ (**h**) $t = 150$ (**i**) $t = 200$

Figure A4. Trajectory of agents (FF-C). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.



(**a**) $t = 5$

(**b**) $t = 10$

(**c**) $t = 20$

(**d**) $t = 30$

(**e**) $t = 40$

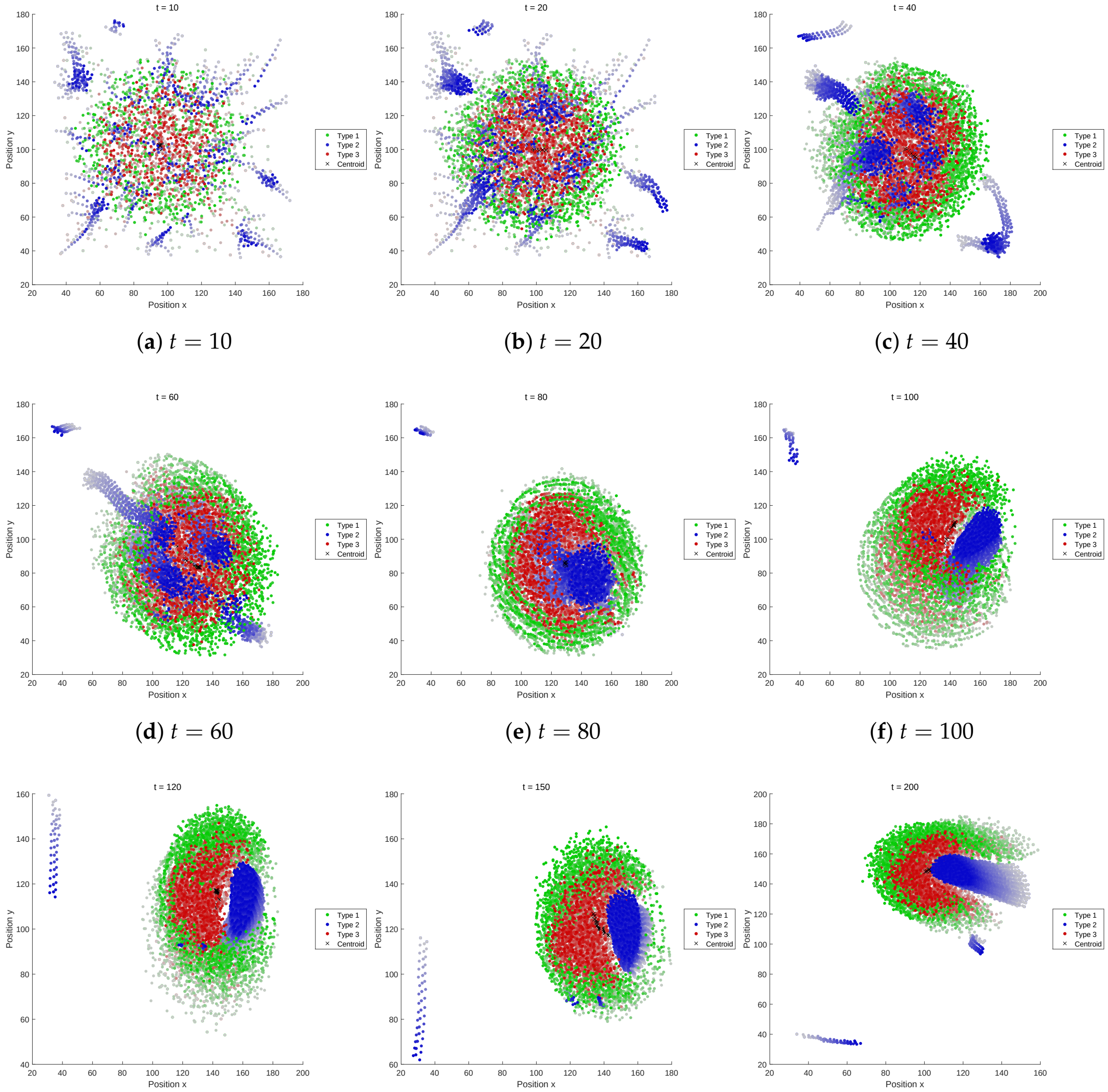
(**f**) $t = 50$

(**g**) $t = 60$

(**h**) $t = 70$

(**i**) $t = 80$

Figure A5. Trajectory of agents (FF-D). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.

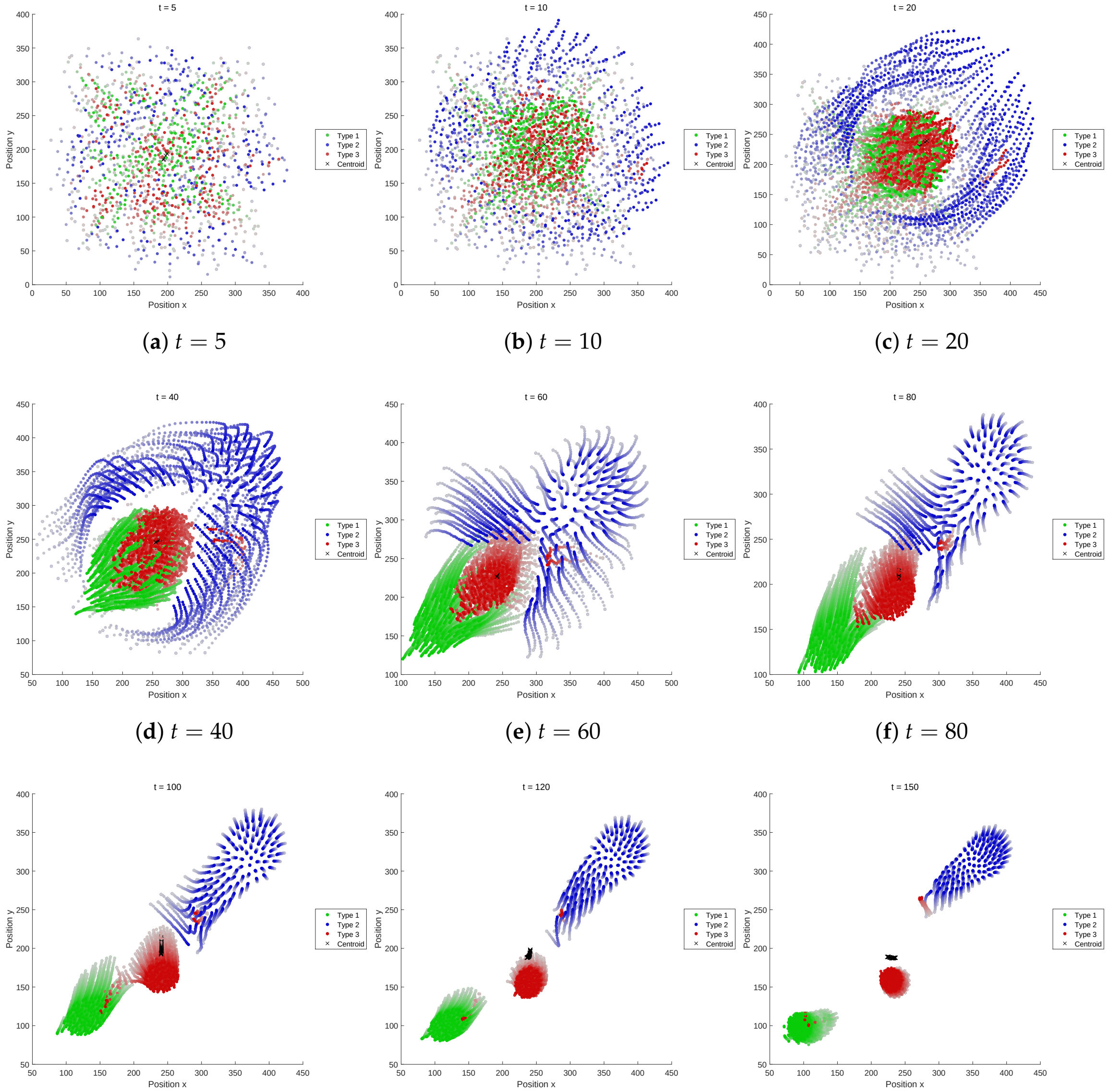


(**a**) $t = 10$ (**b**) $t = 20$ (**c**) $t = 40$

(**d**) $t = 60$ (**e**) $t = 80$ (**f**) $t = 100$

(**g**) $t = 120$ (**h**) $t = 150$ (**i**) $t = 200$

Figure A6. Trajectory of agents (FF-E). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.

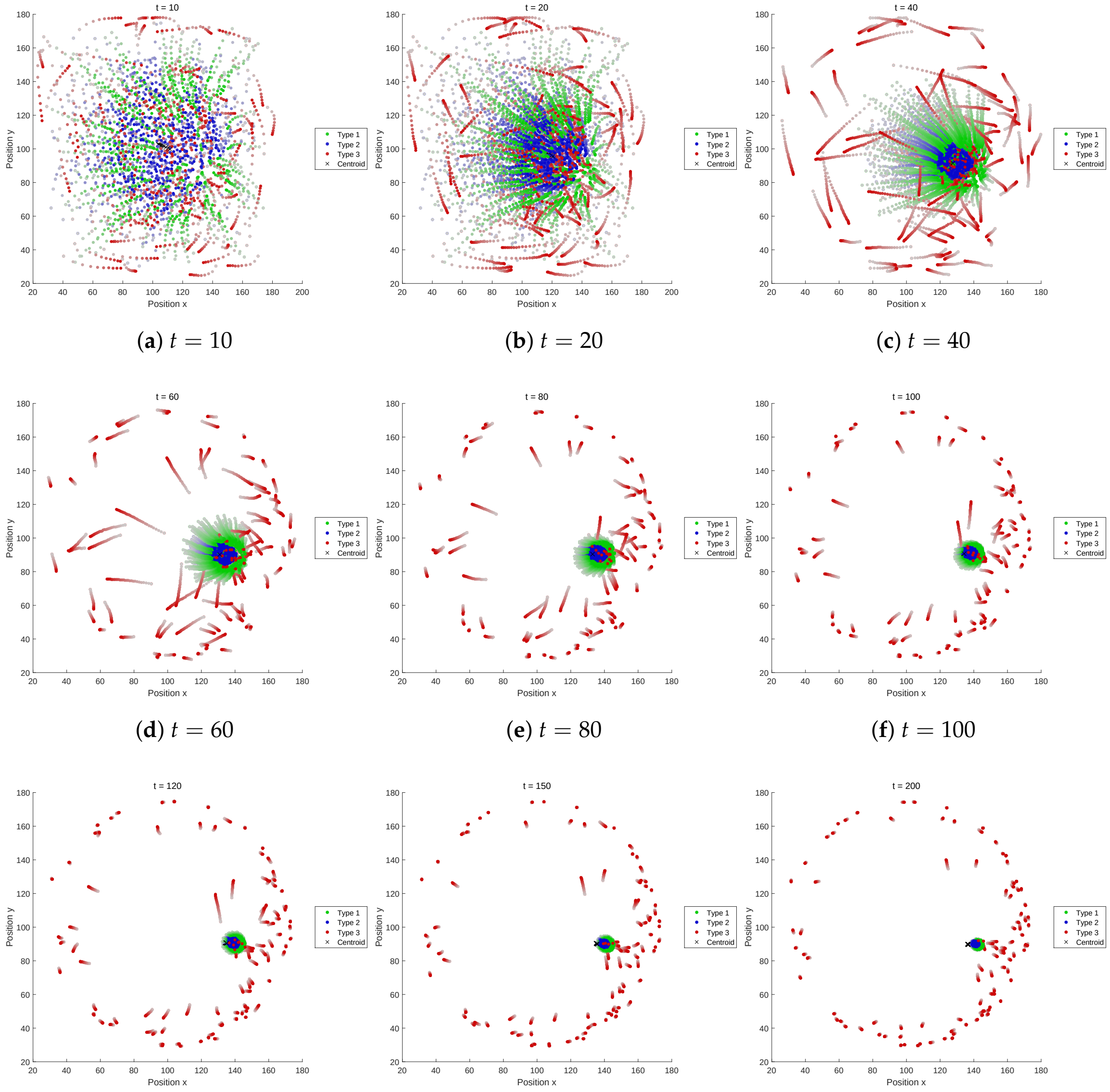


(**a**) $t = 5$ (**b**) $t = 10$ (**c**) $t = 20$

(**d**) $t = 40$ (**e**) $t = 60$ (**f**) $t = 80$

(**g**) $t = 100$ (**h**) $t = 120$ (**i**) $t = 150$

Figure A7. Trajectory of agents (FF-F). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.



(**a**) $t = 10$ (**b**) $t = 20$ (**c**) $t = 40$

(**d**) $t = 60$ (**e**) $t = 80$ (**f**) $t = 100$

(**g**) $t = 120$ (**h**) $t = 150$ (**i**) $t = 200$

Figure A8. Trajectory of agents (FF-G). The semi-transparent dots behind the agents indicate their past positions at the recent 60 time steps. The "×" mark traces the trajectory of swarm centroid.

References

1. Vicsek, T.; Czirok, A.; Ben-Jacob, E.; Cohen, I.; Shochet, O. Novel Type of Phase Transition in a System of Self-Driven agents. *Phys. Rev. Lett.* **1995**, *75*, 1226–1229. [CrossRef] [PubMed]
2. Vicsek, T. A question of scale. *Nature* **2001**, *411*, 421.
3. Schmickl, T.; Stefanec, M.; Crailsheim, K. How a life-like system emerges from a simple particle motion law. *Sci. Rep.* **2016**, *6*, 37969. [CrossRef] [PubMed]
4. Reynolds, C. Flocks, Herds, and Schools: A Distributed Behavioral Model. *Comput. Graph.* **1987**, *25*, 25–34. [CrossRef]
5. Murphy, R.R. *Disaster Robotics*; MIT Press: Cambridge, MA, USA, 2014.
6. Doursat, R.; Sayama, H.; Michel, O. *Morphogenetic Engineering*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 191–208.
7. Sayama, H. Analysis and design of self-organizing heterogeneous swarm systems. In Proceedings of the 2016 Conference on Complex Systems, Amsterdam, The Netherlands, 19–22 September 2016.

8. Doursat, R.; Sayama, H.; Michel, O. A review of morphogenetic engineering. *Nat. Comput.* **1987**, *12*, 517–535. [CrossRef]
9. Sayama, H. Robust Morphogenesis of Robotic Swarms. *IEEE Comput. Intell. Mag.* **2010**, *5*, 43–49.
10. Sayama, H. Swarm chemistry. *Artif. Life* **2009**, *15*, 105–114. [CrossRef] [PubMed]
11. Sayama, H.; Wong, C. Quantifying evolutionary dynamics of Swarm Chemistry. In Proceedings of the Eleventh European Conference on Artificial Life (ECAL 2011), Paris, France, 8–12 August 2011; pp. 729–730.
12. Sayama, H. Morphologies of self-organizing swarms in 3D swarm chemistry. In Proceedings of the Fourteenth InternationAl Conference on Genetic and Evolutionary Computation Conference, Philadelphia, PA, USA, 7–11 July 2012; pp. 577–584.
13. Sayama, H. Swarm Chemistry Evolving. In Proceedings of the Alife XII Conference, Odense, Denmark, 19–23 August 2010; pp. 32–33.
14. Sayama, H. Swarm Chemistry Homepage. Available online: http://bingweb.binghamton.edu/~sayama/SwarmChemistry/ (accessed on Apr. 29, 2018).
15. Sayama, H. Seeking Open-Ended Evolution in Swarm Chemistry. In Proceedings of the IEEE ALIFE 2011, Paris, France, 11–15 April 2011; pp. 186–193.
16. Sayama, H. Guiding Designs of Self-Organizing Swarms: Interactive and Automated Approaches. In *Guided Self-Organization: Inception*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 365–387.
17. Nishikawa, N.; Suzuki, R.; Arita, A. Coordination Control Design of Heterogeneous Swarm Robots by means of Task-Oriented Optimization. *Artif. Life Robot.* **2016**, *21*, 57–68. [CrossRef]
18. Sayama, H. Four Classes of Morphogenetic Collective Systems. In Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14), New York, NY, USA, 31 July–2August 2014.
19. Sayama, H. Behavioral Diversities of Morphogenetic Collective Systems. In Proceedings of the European Conference on Artificial Life 2015, New York, NY, USA, 20–24 July 2015.
20. Eshelman, L.J.; Schaffer, J.D. Real-coded Genetic Algorithms and Interval-Schemata. *Found. Genet. Algorithms* **1993**, *2*, 187–202.
21. Palmer, A.R. Animal Asymmetry. *Curr. Biol.* **2009**, *19*, 473–477. [CrossRef] [PubMed]
22. Palmer, A.R. Symmetry breaking and the evolution of development. *Science*, **2004**, *306*, 828–833. [CrossRef] [PubMed]
23. Castellani, E.; Ismael, J. Which Curie's Principle? *Philos. Sci.* **2016**, *83*, 1002–1013. [CrossRef]
24. Chalmers, A.F. Curie's Principle. *Br. J. Philos. Sci.* **1970**, *21*, 133–148. [CrossRef]
25. Kano, T.; Osuka, K.; Kawakatsu, T.; Matsui, N.; Ishiguro, A. A Minimal Model of Collective Behaviour Based on Non-reciprocal Interactions. In Proceedings of the fourteenth European Conference on Artificial Life (ECAL 2017), Lyon, France, 4–8 September 2017; pp. 237–244.