Pattern Formation in Networks Inspired by Biological Development

Kei Ohnishi, Kaori Yoshida, and Mario Köppen

Kyushu Institute of Technology,
680-4 Kawazu, Iizuka, Fukuoka 820-8054 Japan
ohnishi@cse.kyutech.ac.jp, kaori@ai.kyutech.ac.jp,
mario@ndrc.kyutech.ac.jp

Abstract. We present a system for pattern formation in networks that is inspired by biological development. The system hierarchically forms a specific pattern in a rough-to-detail manner as seen in biological development. We apply the system to generating gray-scaled images. In this application, a network topology is first generated from a given original image in which pixels correspond to network nodes and the nodes are linked to one another in a certain way, and then the system is run in the generated network. The pattern formed by the system within the network is a distribution of brightness of pixels which are equal to nodes. The system is shown to create a variety of images which are influenced by the structure in the original image in a rough-to-detail manner. This feature would be useful for designing patterns under some structural constraints.

Keywords: Biological development, pattern formation, feedback system, image generation, network topology.

1 Introduction

Recently, many proposed engineering methods have been inspired by biological behavior and structures, such as evolutionary computation [1], neural networks, artificial immune systems [2], ant colony systems [3], and so on. The reason is that we can expect to obtain useful hints on creating new methods from the study of biological behavior and structures.

Some of these methods are equivalent to the approaches used in artificial life (A-Life). These approaches create macroscopic structures or functions by only using local interaction between their elements. In these approaches, the macroscopic structures or functions are often quantitatively undefined but qualitatively defined. Consequently, we need to adjust the local interaction rules for our purpose. However, the A-Life approaches have several advantages. One advantage is that these approaches have the possibility to realize structures or functions that are barely realized by a centralized control system.

Simple systems using the A-Life approaches are feedback systems such as cellular automata [7] and L-systems [4][6]. These systems can create complex structures from a simple initial state. Complex structures are created by repeatedly applying the rules of rewriting symbols corresponding to their structures. It

F. Huang and R.-C. Wang (Eds.): ArtsIT 2009, LNICST 30, pp. 64–71, 2010.

is simple but difficult task for us to estimate the final structure from the initial state and rules because the rules are applied locally. Their effective use may be to discover or to create interesting structures by adjusting the rules and the initial state by trial and error. Interesting structures that we could have never imagined may be discovered by chance.

Following this idea, we previously proposed a new feedback system inspired by biological development, which we referred to as the "BiDevelopment System" [5]. The BiDevelopment System is supposed to form patterns in Euclidean spaces. In the present paper, we present a new algorithm by extending an example algorithm of the BiDevelopment System. The algorithm is able to form patterns in networks. In addition, we apply the algorithm to generating gray-scaled images.

The advantage of the BiDevelopment System is the embedded rough-design structure and various possible generated structures under the given rough-design structure. Conventional feedback systems, such as the L-System or cellular automaton, apply only one rule to each element at each feedback loop. It is difficult to design an entire final structure from the beginning and embed it into the system. Since each module of the proposed system determines final detail from the basic structure, it is easier to embed the rough-design structure before the system runs.

The present paper is organized as follows. Section 2 describes the BiDevelopment System. In Section 3, we present the algorithm of the BiDevelopment system for pattern formation in networks and also apply the algorithm to generating gray-scale images. Section 4 describes our conclusion and future work.

2 Pattern Formation in Euclidean Spaces

The new algorithm, which will be presented in Section 3, is realized based on an example algorithm of the recently proposed BiDevelopment System. So, we recall the BiDevelopment System in this section. The hints for the BiDevelopment System, which is a mechanism of biological development, are described in [5]. A biological development process forms an adult body from a mother cell based only on design information of the mother cell in a rough-to-detail manner. In this process, proteins hierarchically diffuse among all or some cells and gives them positional information, telling the cells what organs they will eventually become.

2.1 The Conceptual Framework

The BiDevelopment System outputs values on a space, which is its input. Values of the system parameters are provided in a character code. It has an initialization module and three major modules. The module numbers like (0), (1) etc below correspond to numbers in Figure 1. The initialization module (0) defines the area where system outputs exist. The module (1) generates the global positional information that determines what each element processes. The module (2) lets each element autonomously behave based on its positional information. Local mutual interaction among elements may take place. The module (3) exchanges the output of the module (2) into the output range of elements. The final output value of each element is determined by the feedback module (4b).

The module that generates data related with structure is the module (3). The values that the feedback process at the module (3) generates after certain number of feedback loops determine the final structure of the system output. The modules (1) and (2) sequentially generate the structure of the final output pattern from the BiDevelopment System.

The modules (1) and (3) include feedback modules (4a) and (4b), respectively. The feedback module (4a) generates the positional information of the next state from that of the previous state. The feedback module (4b) gradually narrows the range of output values of each feedback process.

2.2 Example Algorithm

An example algorithm under the framework of the BiDevelopment generates values of y for given coordinates of x. Figure 1 shows the flow of modules in the algorithm.

The task here is to draw a figure in a (x, y) space by generating values of y for coordinates of x in a fixed closed area, $x \in [x_1, x_n]$ $(x_1 < x_n)$. Then, the modules of the algorithms are as follows. The modules are (0), (1), (2), (3), (4a), and (4b) in Figure 1. A character code presented in Figure 1 encodes values of the algorithm parameters.

(0) initialization
The module (0) in Figure 1 inputs the range of x space where y values are generated to draw a figure. The initial data inputted here are $(x_1,\ x_n)$.

(1) generating positional information
The module (1) in Figure 1 gives positional information of each x coordinate using diffusion functions such as Equation (1) (see Figure 2(a) (1).)

$$M(x, A_k, \mu_k, \sigma_k) = M_k(x) = A \exp^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}} \tag{1}$$

Let us call the $M_k(x)$ in Equation (1) a diffusion function from the analogy to diffusing protein among cells in biological development. The iteration number of feedback is represented by k. Any function is available for the diffusing functions, and we adopt a Gaussian function for M_k in this paper. The positional information is determined by the $M(x, A_k, \mu_k, \sigma_k)$ whose parameters, A_k, μ_k, and σ_k are read from the character code. Although the algorithm presented here uses only one diffusion function as in Equation (1), other algorithms may use multiple diffusion functions. That is, $\sum_i M(x, A_{ki}, \mu_{ki}, \sigma_{ki})$ is used.

(2) gene expression
The module (2) in Figure 1 divides x into several areas according to the s_i read from the character code and values of $M_k(x)$ (see Figure 2(a) (2)). Labels are

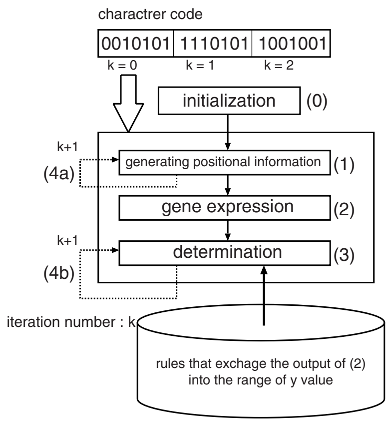


Fig. 1. Process flow of the algorithm

given to the divided areas of x. We compared the given labels to the genetic information and named this module as a *gene expression*.

(3) determination
The module (3) in Figure 1 converts the group labels outputted from the previous module (2) together with previous y range into the range of values of y (see Figure 2(a) (3)). The conversion rules that are different in each processing time, k, must be prepared.

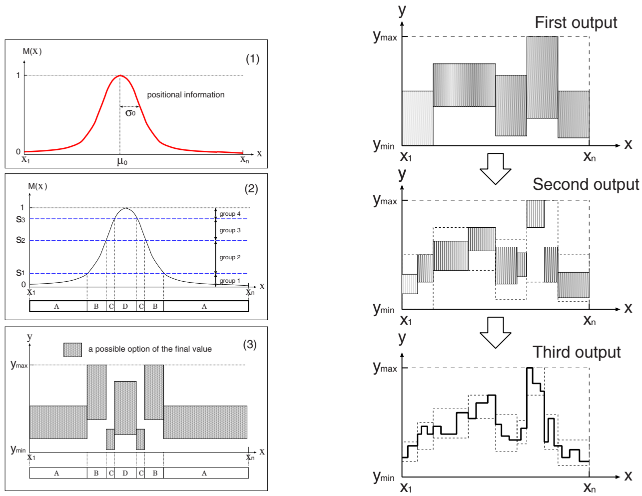
(4) feedback
These modules (4a) and (4b) in Figure 1 are feedback modules at the modules (1) and (3), respectively.

(4a) feedback for the module (1)
The module (4a) calculates μ_k for diffusion function $M_k(x)$ from $M_{k-1}(x)$. It first reads m from the character code, substitutes the m for Equation (2), and determines μ_k, where μ_0 is given in the character code.

$$\mu_k = x, if \quad M_{k-1}(x) < m < M_{k-1}(x+1), or \qquad (2)$$
$$if \quad M_{k-1}(x+1) < m < M_{k-1}(x).$$

When multiple μ_{ki} are obtained for the given m, $\sum_i M(x, A_k, \mu_{ki}, \sigma_k)$ is used in the module (1). Other necessary parameters for Equation (1), A_k and σ_k, are read from the character code.

(4b) feedback for the module (3)
According to the feedback module (4b), the range of y is gradually reduced (see Figure 2(b)). The rules that determine the new ranges of y from the past ranges y and group labels must be previously prepared in a rule-base at the module (3).



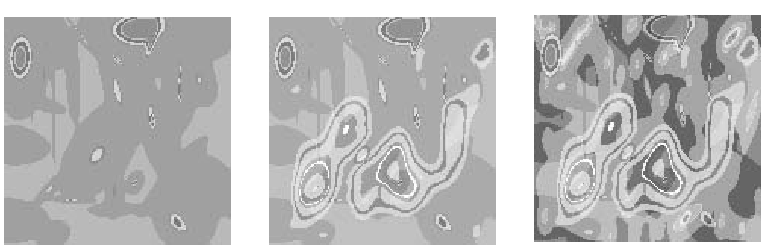
(a) Example outputs of the first processing, $k = 0$

(b) Example outputs of the module (3)

Fig. 2. Example outputs

2.3 Generating Gray-Scaled Images

In this section, we demonstrate how the BiDevelopment System creates gray-scaled images. The algorithm used here is similar to the algorithm in Section 2.2 except for a 2-D target space, (x_1, x_2), and the initial diffusion function combining 50 Gaussian functions. The algorithm used here obtains a final output just after the third feedback. Images are displayed by converting the generated y values to the brightness of pixels. An example of generated images is shown in Figure 3. In Figure 3, outputs at each feedback are shown. Although outputs



(a) The output at $k = 1$ (b) The output at $k = 2$ (c) The final output at $k = 3$

Fig. 3. Example gray-scaled images generated by the BiDevelopment System

except the final output represent ranges of z values on the x-y plane, we create images from those outputs by using the average of the ranges of z values.

3 Pattern Formation in Networks

In the image generation shown in Section 2.3, we first considered correspondence between discrete coordinates in a two dimensional Euclidean space (a x-y plane) and pixels in a gray-scaled image, and then, we generated gray-scaled images by using z values that the algorithm generated in the x-y plane as brightness of the pixels in the images. Therefore, the generated images basically, as shown in Figure 3, include shape of cross section between multiple Gaussian functions and a plane parallel to the x-y plane.

One of the ways to create more various images is to change a field or a space in which the algorithm forms patterns. So, in this section, we extend the example algorithm of the BiDevelopment System presented in Section 2.3 to the one which is able to form patterns in networks. To form patterns in a network with the extended algorithm, the algorithm has to prepare a diffusion function for a network. A diffusion function used herein is represented as Equation (3). The diffusion function is generated at some node.

$$M(h_s, A_k, \mu_k, \sigma_k) = M_k(h_s) = A \exp^{-\frac{(h_s-\mu_k)^2}{2\sigma_k^2}}, \tag{3}$$

where h_s is the minimal number of hops from a node of focus to the node, at which the diffusion function is generated. The diffusion function is basically the same as Equation (1) shown in Section 2.2, but x in Equation 1 is changed to h_s in Equation (3).

In addition, we apply the extended algorithm to generating gray-scaled images. In this application, a network is first generated from a given original image, in which pixels correspond to network nodes and the nodes are linked to one another in a certain way (see Figure 4), and then the algorithm is run in the generated network. The formed pattern in the network is a distribution of brightness of pixels which are equal to nodes.

How to generate a network from a given image is as follows. Each node, which has a corresponding pixel, makes L links to other nodes. Nodes to which each node makes links are selected from among all of the nodes with probability inversely proportional to difference between brightness of each node and other node. That is, nodes with similar brightness are likely to link to one another. The brightness of the pixels of the give image are used only to generate a network, which becomes a field, to which the algorithm is applied. In the image generation here, we set L to be 6.

The algorithm here uses the initial diffusion function combining 50 Gaussian functions, and obtains a final output just after the third feedback. Figure 5 shows three examples of the generated images and the original image used for generating the network. In Figure 5, outputs at each feedback are shown. Although outputs except the final output represent ranges of values (brightness) on the

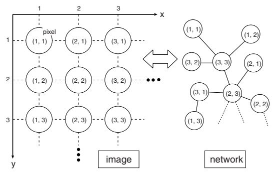
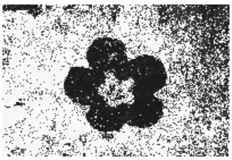
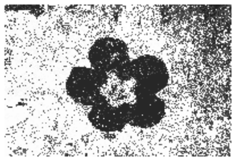


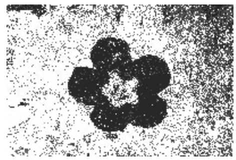
Fig. 4. Correspondence between nodes in a network and pixels in a gray-scaled image



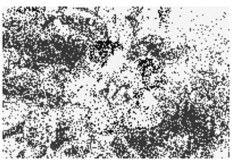
(a) First example. The output at $k = 1$.



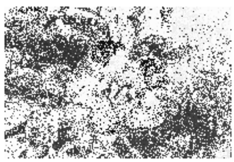
(b) First example. The output at $k = 2$.



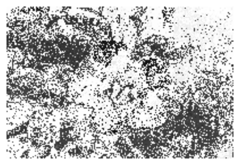
(c) First example. The final output at $k = 3$.



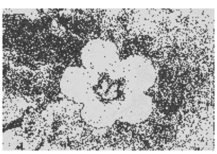
(d) Second example. The output at $k = 1$.



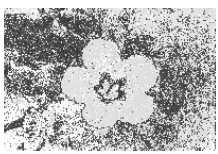
(e) Second example. The output at $k = 2$.



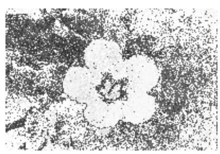
(f) Second example. The final output at $k = 3$.



(g) Third example. The output at $k = 1$.



(h) Third example. The output at $k = 2$.



(i) Third example. The final output at $k = 3$.

(j) Original image.

Fig. 5. Three examples of gray-scaled images generated by the extended algorithm with three different character codes

network, we create images from those outputs by using the average of the ranges of values.

Figure 5 suggests that the algorithm presented in this section can create various images. In the application here, we generated the network using the distribution of brightness of pixels in the original image, so that the images generated by the algorithm seem to be influenced by the distribution of brightness.

4 Concluding Remarks

In this paper we realized the new algorithm under the framework of the recently proposed BiDevelopment System. The realized algorithm is meant to form patterns in networks. In addition, the algorithm was applied to generating gray-scaled images. In this application, a network was first generated based on the given image and then the algorithm was run in the network. The results of the image generation suggested that the algorithm can create various images which are influenced by the distribution of brightness of an original image, which is used for generating a network.

As shown in this paper, it would be possible to represent fundamental desired structures or structural constraints as a form of network topologies. So, if we combine the use of network topologies, the strategy of rough-to-detail pattern formation of the BiDevelopment System, and an optimization technique which adjusts the parameter of the BiDevelopment System based on human evaluation, the BiDevelopment System might be able to be an emergent design tool that is handleable for humans. In the future work, we will tackle with this problem.

Acknowledgment

This work was supported by the Japan Society for the Promotion of Science through a Grant-in-Aid for Scientific Research (S)(18100001).

References

1. Bäck, T.: Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press, Oxford (1996)
2. Dasgupta, D. (ed.): Artificial immune systems and their applications. Springer, Heidelberg (1999)
3. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: optimization by a colony of cooperating agents. IEEE Trans. on Systems, Man, and Cybernetics-Part B 26(2), 29–41 (1996)
4. Lindenmayer, A.: Mathematical models for cellular interaction in development, PartI and PartII. Journal of Theoretical Biology 18, 280–315 (1968)
5. Ohnishi, K., Takagi, H.: Feed-Back Model Inspired by Biological Development to Hierarchically Design Complex Structure. In: IEEE International Conference on System, Man, and Cybernetics (SMC 2000), Nashville, TN, USA, October 8-11, pp. 3699–3704 (2000)
6. Prusinkiewicz, P., Lindenmayer, A.: The algorithmic beauty of plants. Springer, New York (1990)
7. Wolfram, S.: Cellular automata and complexity: collected papers. Addison-Wesley, Reading (1994)