Square Kufic Pattern Formation by Asynchronous Cellular Automata

Seyyed Amir Hadi Minoofam[1] and Azam Bastanfard[2]

[1] Computer Engineering Faculty, Islamic Azad University-Qazvin Branch,
Daneshgah St., Nokhbegan Blvd., Qazvin, Iran
[2] Computer Engineering Faculty, Islamic Azad University-Karaj Branch,
Imam Ali Complex, Moazen Blvd., Karaj, Iran
`minoofam@qiau.ac.ir, bastanfard@kiau.ac.ir`

Abstract. Algorithms Design for pattern formation is an interesting science and many investigations are done about it. In this paper, a novel pattern formation method using asynchronous cellular automata for generating square Kufic patterns is explored. Square Kufic is a kind of script that is used in many Islamic architecture buildings. In our new method, cellular automata rules causing a specified kind of pattern are manually extracted. The implementation results show this fact by using some special patterns in replicators.

Keywords: Asynchronous Cellular Automata, Computer Graphics, Pattern Formation, Square Kufic script.

1 Introduction

Cellular automata are dynamical systems such that the current state of each cell is determined by its neighborhood state at the previous time step. We consider in this paper an interactive particle system modeled by a cellular automaton having two kinds of cells, occupied and empty cells.

The main contribution of this paper is using asynchronous cellular automata to generate script patterns. These particular patterns are used in cultural heritage. Square Kufic patterns are based on grids and the best way for implementing such patterns is a grid based method like cellular automata [1]. Using cellular automata has several benefits. For instance, it is simple because it is based on square grids like the forming elements of square Kufic script. Also it is scalable and flexible.

The rest of the paper is organized as follows. Section 2, describes the basic notions of asynchronous two dimensional cellular automata. Section 3, provides the rule set for generating some patterns. Experimental results appear in section 4. The paper is concluded in section 5.

2 Asynchronous Cellular Automata

In general, it has been found that the asynchronous cellular automata evolve much differently from their synchronous counterparts [2]. There are various methods to update cells asynchronously and these fall into two distinct categories, step-driven and time-driven. In this paper, we use the latter method.

This paper uses extended Moore neighborhood in its simplest form. It simply handles joint and disjoint letters. In some literatures the beginning letters of the direction names are used to refer neighbor cells. For example "C" points to the internal cell and "NE" refers to the northeast cell. We extend this method of naming to refer to the cells in an easy understanding way [3]. For instance, "NEE" refers to the northeast east cell, which means one hop to north and then two hops toward east direction.

Next section explains about transition rules for generating square Kufic patterns.

3 Rule Extraction

Each cell's state is assumed to take an integer equal either to nil or unity. The nil state denotes an empty cell having white color, and the other state is one occupied by a particle having black color.

Figure1 (a) and (b) show two architectural buildings contain two models of "Allah" pattern.

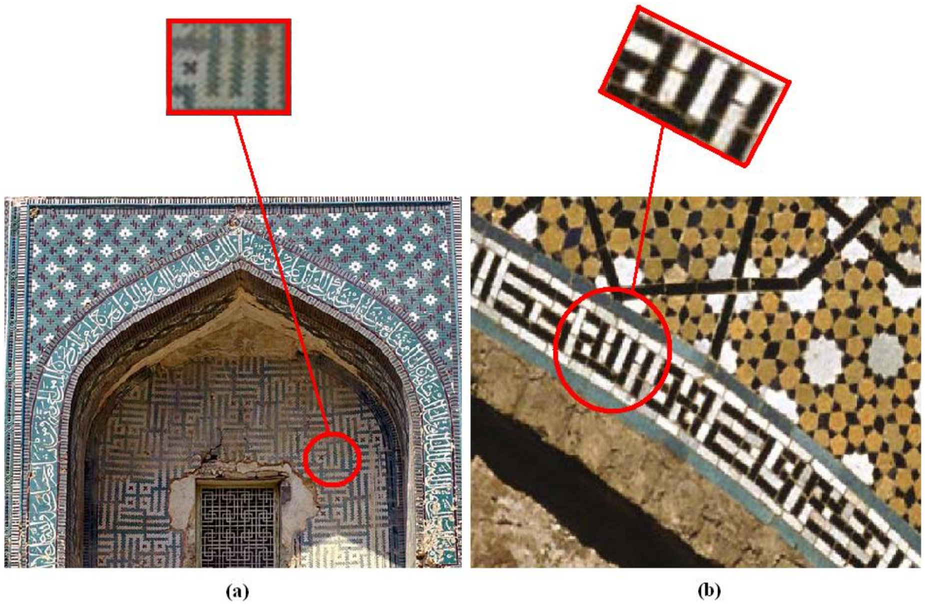

Fig. 1. (a) Model 1 of "Allah" pattern tiling from the Tughlugh Temur Mausoleum of Xinjiang. (b) Model 2 of "Allah" pattern tiling from the Imam Mosque of Isfahan.

First of all, the patterns have to decompose in to their constructing letters. Then, the transition rules set for each letter must be extracted manually. After that, the cellular automata can produce the whole of each pattern asynchronously.

In each rule if all the mentioned cells are black, then the rule is true, and it causes an effect in cellular automata at the next time step. Otherwise, if even one of the mentioned cells isn't black, then the rule is false which causes no effect on the cellular grid.

Figure 2 demonstrates the asynchronous process for generating patterns respectively.

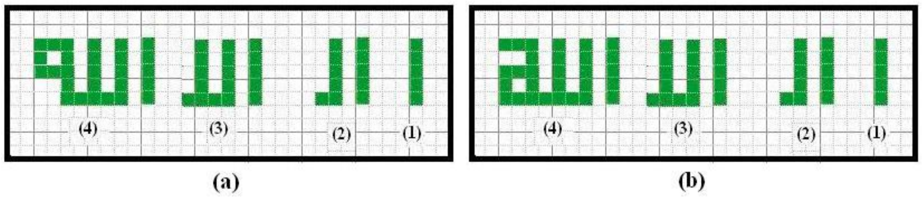

Fig. 2. (a) Generating process for model 1 of "Allah" pattern. (b) Generating process for model 2 of "Allah" pattern.

Consequently, it is possible to provide a flexible algorithm for generating these patterns. Reused rules can make the algorithm shorter and simpler too [4].

In this step, we define a transition function φ that operates on the rules with xor "$\oplus$" operation.

$$\varphi[R_1, R_2, ..., R_n] = \oplus^n{}_{i=1} R_i \tag{1}$$

Relation (1) describes that all rules R_i should be xor ($\oplus$) with each other to introduce the correct results. Note that for each letter, some functions φ with different R_is are existing.This Algorithm is the determination of each transition that is a mapping from a neighborhood pattern to anther neighborhood pattern. We at first set all of the transitions to be "inactive": Any patterns of neighborhood are not changed by the update. In the following we will replace some inactive transitions with "active" ones that change the pattern of neighborhood by the update [5]. Note that, for the simplicity sake, in this section we were considering only a few set rules.

The next section demonstrates some results that are obtained using above algorithm.

4 Results

The results of our algorithms which are implemented in Visual C++ environment using OpenGL library are illustrated as follows. An example of executing on model 1 of "Allah" pattern is illustrated in figure 3 (a). In figure 6 (b), the execution of relation on model 2 of this pattern is shown. Result of executing relation on model 1 of "Allah" pattern is illustrated in figure 3 (c). In figure 3 (d), the execution on model 2 of this pattern is demonstrated.

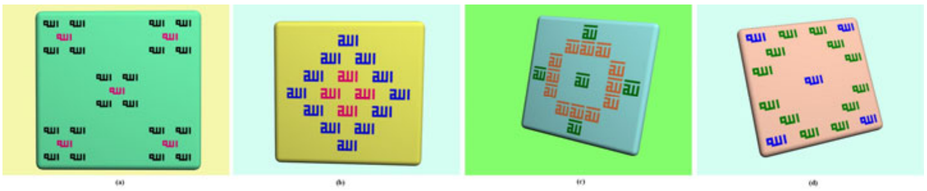

Fig. 3. (a) Applying on model 1 of "Allah" pattern after 40 steps. (b) Applying on model 2 of "Allah" pattern after 24 steps. (c) Applying on model 2 of "Allah" pattern after 24 steps. (d) Applying on model 1 of "Allah" pattern after 24 steps.

5 Conclusion

Using asynchronous cellular automata leads to efficient implementation of the algorithm. Because transition rules become active in a limited time steps, the algorithm is very simple. Also, various patterns can be generated without any rule confliction. Moreover, common parts in each letter will be formed by using the same rules. A comparison between this paper algorithm and other algorithms may be expected, but no other method was investigated before. Therefore, this is the first algorithm for generating square Kufic patterns by asynchronous CA.

The Square Kufic script is a good instance in calligraphy which provides different versions of writing holly words. As this script is based on a square grid, the proposed algorithm not only provides the simplicity and flexibility for generating these patterns but also has the power for controlling complexity and possibility of extending the functionality of the rules. Examples of Square Kufic script can be found in cultural heritage and old literatures. Also it is so popular in modern architectures, handicrafts, decorations and art. One of our future works is designing more complex patterns and searching about making use of geometrical transformations in cellular automata.

References

1. Steeb, W.H.: The Nonlinear Workbook. World Scientific Publishing, Singapore (2005)
2. Beigy, H., Meybodi, M.R.: Asynchronous Cellular Learning Automata. J. Auomatica 44(4), 1350–1357 (2008)
3. Minoofam, S.A.H., Bastanfard, A.: A Novel Algorithm for Generating Muhammad Pattern Based on Cellular Automata. In: 13th WSEAS International Conference on Applied Mathematics (MATH 2008), pp. 339–344 (2008)
4. Mitra, S., Kumar, S.: Fractal Replication in Time-manipulated One Dimensional Cellular Automata. J. Complex Systems 16(3), 191–207 (2006)
5. Schiff, J.L.: Cellular Automata: A Discrete View of the World. John Wiley, Chichester (2008)