Recognizing Complex Behavior Emerging from Chaos in Cellular Automata

Gabriela M. González[1(✉)], Genaro J. Martínez[1,2], M. A. Aziz Alaoui[3], and Fangyue Chen[4]

[1] Artificial Life Robotics Lab, Escuela Superior de Cómputo, Instituto Politécnico Nacional, Mexico City, Mexico
gmorenog1400@alumno.ipn.mx
[2] Unconventional Computing Lab, University of the West of England, Bristol BS16 1QY, UK
genaro.martinez@uwe.ac.uk
[3] Normandie Univ, UNIHAVRE, LMAH, FR-CNRS-3335, ISCN, BP. 540, 76600 Le Havre, France
aziz.alaoui@univ-lehavre.fr
[4] School of Science, Hangzhou Dianzi University, Hangzhou, China
fychen@hdu.edu.cn

Abstract. In this research, we explain and show how a chaotic system displays non-trivial behavior as a complex system. This result is reached modifying the chaotic system using a memory function, which leads to a new system with elements of the original function which are not evident in a first step. We proof that this phenomenology can be apprehended selecting a typical chaotic function in the domain of elementary cellular automata to discover complex dynamics. By numerical simulations, we demonstrate how a number of gliders emerge in this automaton and how some controlled subsystems can be designed within this complex system.

Keywords: Complex dynamics · Chaos · Emergence · Gliders
Glider guns · Memory

1 Preliminaries

As far as we know a classification scheme of complex systems is missing, although, in the case of elementary cellular automata, several approaches have been described in recent decades. Nevertheless, this is a difficult problem that is initially determined as an undecidable problem in the context of the theory of cellular automata [9]. However, several approaches have been considered (we can refer for example to [1,3,12,17,31] and references cited therein). These researches show that, to date, all approaches using elementary cellular automata, do not match, for more details, see [14].

In this paper, we study a classic chaotic elementary cellular automaton and, using a memory function, we describe elements of non-trivial behaviour which

A. J. Morales et al. (Eds.): ICCS 2018, SPCOM, pp. 82–90, 2018.
https://doi.org/10.1007/978-3-319-96661-8_8

emerge as patterns. The last are named gliders, particles, waves, or mobile self-localizations. This non-trivial behaviour is qualified as *complex*. Specialized and historic books recommended to introduce to complex systems theory are [4–6,19,21]. For cellular automata theory, we refer to [1,18,27,29].

Here, we study a particular case with elementary cellular automaton rule 126, classified by Wolfram as a chaotic rule in [29,30]. Reviewing and composing the original function with a memory function we will discover non-trivial patterns. The original study about this analysis was published in [15]. This way, the present paper is an extension on its controllability and codification of gliders in the evolution rule with memory and how scale complex its behaviour by glider collisions. The paper has the next structure. Section two introduces basic concepts. Section three displays a description of rule 126 and its chaotic behaviour. Section four explains how works a memory function in cellular automata. Section five shows the non-trivial behaviour in rule 126 with memory.

1.1 Basic Notation

One-dimensional cellular automata is represented by an array of *cells* x_i where $i \in Z$ and each x takes a value from a finite alphabet Σ. Thus, a sequence of cells $\{x_i\}$ of finite length n describes a string or *global configuration* c on Σ. The set of finite configurations will be expressed as Σ^n. An evolution is comprised by a sequence of configurations $\{c_i\}$ produced by the mapping $\Phi : \Sigma^n \rightarrow \Sigma^n$; thus the global relation is symbolized as: $\Phi(c^t) \rightarrow c^{t+1}$, where t represents time and every global state of c is defined by a sequence of cell states. The global relation is determined over the cell states in configuration c^t updated at the next configuration c^{t+1} simultaneously by a local function φ as follows: $\varphi(x^t_{i-r}, \ldots, x^t_i, \ldots, x^t_{i+r}) \rightarrow x^{t+1}_i$.

Wolfram represents one-dimensional cellular automata with two parameters (k, r), where $k = |\Sigma|$ is the number of states, and r is the neighbourhood radius. This way, elementary cellular automata domain is defined by parameters $(2, 1)$. There are Σ^n different neighbourhoods (where $n = 2r + 1$) and k^{k^n} distinct evolution rules. The evolutions in this paper have periodic boundary conditions.

Conventional cellular automata are ahistoric (memoryless): i.e., the new state of a cell depends on the neighbourhood configuration solely at the preceding time step of φ. Thus, cellular automata with *memory* can be considered as an extension of the standard framework of cellular automata where every cell x_i is allowed to remember some period of its previous evolution. Basically memory is based on the state and history of the system, thus we design a memory function ϕ, as follows: $\phi(x^{t-\tau}_i, \ldots, x^{t-1}_i, x^t_i) \rightarrow s_i$, such that $\tau < t$ determines the backwards degree of memory and each cell $s_i \in \Sigma$ is a function of the series of states in cell x_i up to time-step $t - \tau$. Finally to execute the evolution we apply the original rule again as follows: $\varphi(\ldots, s^t_{i-1}, s^t_i, s^t_{i+1}, \ldots) \rightarrow x^{t+1}_i$.

In cellular automata with memory, while the mapping φ remains unaltered, a historic memory of past iterations is retained by featuring each cell as a summary of its previous states; therefore cells *canalize* memory to the map φ. As an example, we can take the memory function ϕ as a *majority memory*: $\phi_{maj} \rightarrow s_i$,

where in case of a tie given by $\Sigma_1 = \Sigma_0$ in ϕ, we shall take the last value x_i. So ϕ_{maj} represents the classic majority function for three variables [20] on cells $(x_i^{t-\tau}, \ldots, x_i^{t-1}, x_i^t)$ and defines a temporal ring before calculating the next global configuration c. In case of a tie, it is allowed to break it in favor of zero if $x_{\tau-1} = 0$, or to one whether $x_{\tau-1} = 1$.

The representation of a elementary cellular automata with memory is given as follows: $\phi_{CARm:\tau}$, where CAR represents the decimal notation of a particular elementary cellular automata rule and m the kind of memory given with a specific value of τ. Thus, the majority memory (maj) working in elementary cellular automaton rule 126 checking tree cells ($\tau = 3$) of history is simply denoted as $\phi_{R126maj:3}$.

Note that memory is as simple as any cellular automata, and that the global behaviour produced by the local rule is totally unpredictable, it can lead to emergent properties and so be complex. Memory functions were developed and extensively studied by Sanz in [24–26]. Memory in elementary cellular automata have been studied, showing its potentiality to report complex behaviour from chaotic systems and beyond in [11,12,16], and recently in [7] authors have included hybrid versions. Thus, we can conjecture that a memory function can report complex behaviour as follows: $f_{chaos}(\phi) \rightarrow f_{complex}$.

2 Elementary Cellular Automaton Rule 126

The local-state transition function φ corresponding to rule 126 displays a high concentration of states 1s. This way, $\varphi_{R126} = \{1 \text{ if } 110, 101, 100, 011, 010, 001; 0 \text{ if } 111, 000\}$.

Rule 126 has a chaotic global behaviour typical from class III in Wolfram's classification [29]. In φ_{R126} we can easily recognize an initial high probability of alive cells, i.e. cells in state '1'; with a 75% to appear in the next time and, complement of only 25% to get a state 0.

Figure 1 shows these cases in typical snapshots of rule 126. Evolving from a single cell in state '1' yield a patters like a Sierpinski triangle (Fig. 1a). From a 50% random initial configuration, we can see an unordered evolution

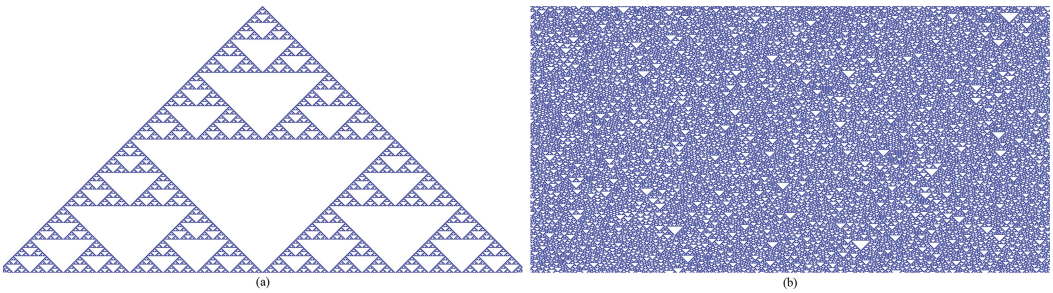

Fig. 1. Dynamics in elementary cellular automaton rule 126. (a) Sierpinski triangle is the evolution in its global dynamics from a single cell in state 1. (b) Second snapshot is calculated from a random initial density to 50%. Both space-time diagrams evolve on a ring of 1,000 cells for 512 generations.

without observing any recognizable pattern (Fig. 1b). Both evolutions induce that evolution rule 126 display a chaotic global behaviour. To explore carefully a lot of these properties we will analyze its basin of attractors in the next section.

2.1 Basins of Attraction

A basin (of attraction) field of a finite cellular automata is the set of basins of attraction into which all possible states and trajectories will be organized by the local function φ. The topology of a single basin of attraction may be represented by a diagram, the *state transition graph*. Thus, the set of graphs composing the field specifies the global behaviour of the system [27].

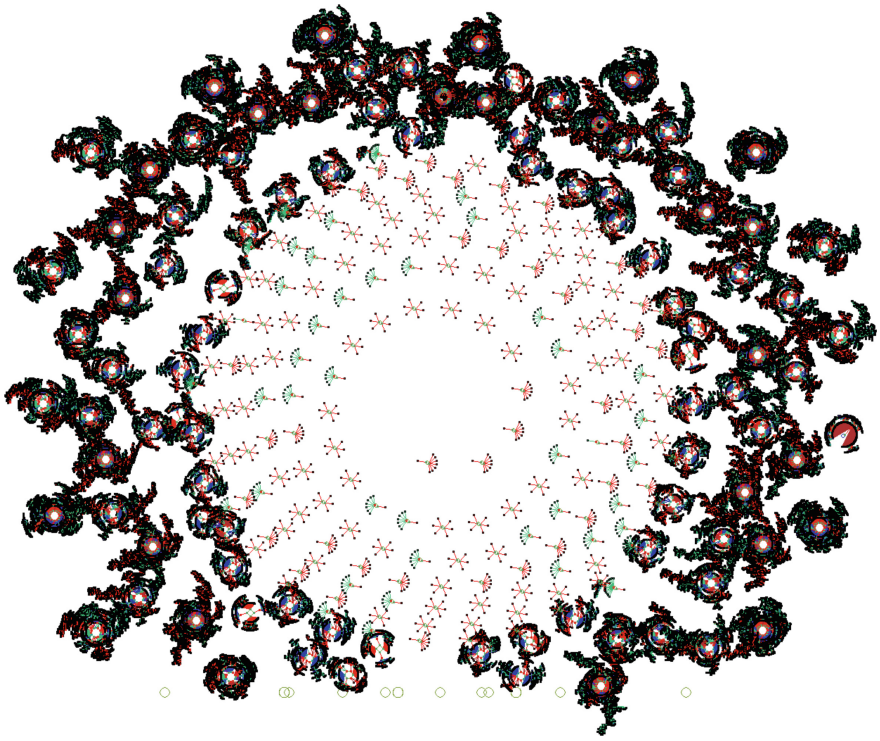

Fig. 2. Basin of attractors for rule 126 and a length of 20 cells.

Generally, a basin can also recognizes cellular automata with chaotic or complex behaviour following previous results on attractors which has been reported by Wuensche and Lesser in [27] for rings of length 2 to 15. Thus, Wuensche characterizes the Wolfram's classes as a basin classification for chaos and complexity in [27].

The basin depicted in Fig. 2 shows the whole set of non-equivalent basins in rule 126 for ring equal to 20 cells.[1] Particularly in this basin we can see that attractors have not long transients or long periodic attractors, but several of them have low in-degree and low leaf density. A quick observation is

[1] Basins and attractors were calculated with *Discrete Dynamical System* DDLab available from http://www.ddlab.org/.

that these basin of attractors are symmetric in their leafs, that induce chaos [8]. Moreover, at the same time we can see some non symmetric attractors and some of them have moderate transients that could induce a non-trivial complex behaviour inside the chaos. A study discussing particularly the complex behaviour by graphs in elementary cellular automata is available in [13].

3 Dynamics Emerging with Memory

This section presents the results of selecting a majority memory (maj) with $\tau = 4$ in rule 126 deriving a new function names $\phi_{R126maj:4}$ [15]. Figure 3 displays an evolution for the rule $\phi_{R126maj:4}$, showing its complex behaviour.

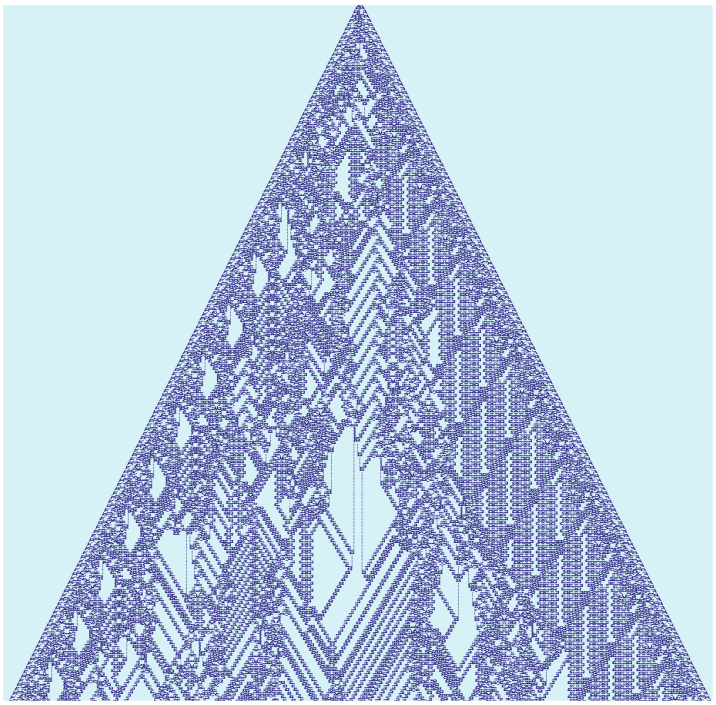

Fig. 3. Complex behaviour emerging in rule 126 with memory, rule $\phi_{R126maj:4}$. Initial configuration 111001 in a ring of 1,000 cells to 1,000 times. The evolution is filtered to get a better view of gliders and collisions. The universe of non-trivial patterns emerge including stationary particles, gliders and several glider guns. Interesting collisions from this initial condition include solitons, annihilations, reflexions, fusions and more. The initial condition was selected intentionally to produce a non-symmetric evolution.

4 Collisions of Gliders

In this section, we have done a systematic analysis of multiple collisions of gliders in $\phi_{R126maj:4}$. The next table presents an equation for every collision and its result. Figure 4 illustrates explicitly these simulations.

Equation	Result	Equation	Result	Equation	Result
$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_1) + e_1(p_1)$	0	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_1) + e_1(p_1)$	s1	$e_1(p_2) + g_1(p_3)^2 + e_1(p_2) + g_2(p_4) + e_1(p_1)$	g_1
$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_2) + e_1(p_2)$	g_1	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_2) + e_1(p_2)$	s_1	$e_1(p_2) + g_1(p_3)^2 + e_1(p_2) + g_2(p_5) + e_1(p_2)$	g_1
$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_3) + e_1(p_2)$	$g_3 + g_3$	$e_1(p_2) + g_1(p_2) + e_1(p_2) + s_1(p_1) + e_1(p_1)$	s_1	$e_1(p_1) + g_1(p_4)^2 + e_1(p_1) + g_2(p_1) + e_1(p_1)$	g_1
$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_4) + e_1(p_1)$	$g_4 + g_4$	$e_1(p_2) + g_1(p_2) + e_1(p_2) + s_1(p_2) + e_1(p_2)$	s_1	$e_1(p_1) + g_1(p_4)^2 + e_1(p_1) + g_2(p_2) + e_1(p_2)$	g_1^2
$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_5) + e_1(p_2)$	g_2	$e_1(p_2) + g_1(p_3) + e_1(p_1) + s_1(p_1) + e_1(p_1)$	s_1	$e_1(p_1) + g_1(p_4)^2 + e_1(p_1) + g_2(p_3) + e_1(p_2)$	$g_3 + g_3$
$e_1(p_2) + g_1(p_2) + e_1(p_2) + g_2(p_1) + e_1(p_1)$	g_2	$e_1(p_2) + g_1(p_3) + e_1(p_1) + s_1(p_2) + e_1(p_2)$	s_1	$e_1(p_1) + g_1(p_4)^2 + e_1(p_1) + g_2(p_4) + e_1(p_1)$	g_2
$e_1(p_2) + g_1(p_2) + e_1(p_2) + g_2(p_2) + e_1(p_2)$	0	$e_1(p_1) + g_1(p_4) + e_1(p_2) + s_1(p_1) + e_1(p_1)$	s_1	$e_1(p_1) + g_1(p_4)^2 + e_1(p_1) + g_2(p_5) + e_1(p_2)$	g_1
$e_1(p_2) + g_1(p_2) + e_1(p_2) + g_2(p_3) + e_1(p_2)$	g_1	$e_1(p_1) + g_1(p_4) + e_1(p_2) + s_1(p_2) + e_1(p_2)$	s_1	$e_1(p_2) + g_1(p_5)^2 + e_1(p_2) + g_2(p_1) + e_1(p_1)$	g_1
$e_1(p_2) + g_1(p_2) + e_1(p_2) + g_2(p_4) + e_1(p_1)$	$g_3 + g_3$	$e_1(p_2) + g_1(p_5) + e_1(p_1) + s_1(p_1) + e_1(p_1)$	s_1	$e_1(p_2) + g_1(p_5)^2 + e_1(p_2) + g_2(p_2) + e_1(p_2)$	g_1
$e_1(p_2) + g_1(p_2) + e_1(p_2) + g_2(p_5) + e_1(p_2)$	$g_4 + g_4$	$e_1(p_2) + g_1(p_5) + e_1(p_2) + s_1(p_2) + e_1(p_2)$	s_1	$e_1(p_2) + g_1(p_5)^2 + e_1(p_2) + g_2(p_3) + e_1(p_2)$	g_1^2
$e_1(p_2) + g1_(p_3) + e_1(p_1) + g_2(p_1) + e_1(p_1)$	$g_4 + g_4$	$e_2(p_5) + g_3(p_2) + e_1(p_2) + g_4(p_2) + e_2(p_5)$	s_2	$e_1(p_2) + g_1(p_5)^2 + e_1(p_2) + g_2(p_4) + e_1(p_1)$	$g_3 + g_3$
$e_1(p_2) + g_1(p_3) + e_1(p_1) + g_2(p_2) + e_1(p_2)$	g_2	$e_1(p_1) + g_3(p_4) + e_2(p_2) + s_2(p_2) + e_2(p_2)$	g_4	$e_1(p_2) + g_1(p_5)^2 + e_1(p_2) + g_2(p_5) + e_1(p_2)$	g_2
$e_1(p_2) + g_1(p_3) + e_1(p_1) + g_2(p_3) + e_1(p_2)$	0	$e_1(p_1) + g_1(p_1)^2 + e_1(p_1) + g_2(p_1) + e_1(p_1)$	g_2	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_1) + e_1(p_1) + g_2(p_1) + e_1(p_1)$	s_1
$e_1(p_2) + g_1(p_3) + e_1(p_1) + g_2(p_4) + e_1(p_1)$	g_1	$e_1(p_1) + g_1(p_1)^2 + e_1(p_1) + g_2(p_2) + e_1(p_2)$	g_1	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_1) + e_1(p_1) + g_2(p_2) + e_1(p_2)$	s_1
$e_1(p_2) + g_1(p_3) + e_1(p_1) + g_2(p_5) + e_1(p_2)$	$g_3 + g_3$	$e_1(p_1) + g_1(p_1)^2 + e_1(p_1) + g_2(p_3) + e_1(p_2)$	g_1	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_1) + e_1(p_1) + g_2(p_3) + e_1(p_2)$	s_1
$e_1(p_1) + g_1(p_4) + e_1(p_1) + g_2(p_1) + e_1(p_1)$	$g_3 + g_3$	$e_1(p_1) + g_1(p_1)^2 + e_1(p_1) + g_2(p_4) + e_1(p_1)$	g_1^2	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_1) + e_1(p_1) + g_2(p_4) + e_1(p_1)$	s_1
$e_1(p_1) + g_1(p_4) + e_1(p_1) + g_2(p_2) + e_1(p_1)$	$g_4 + g_4$	$e_1(p_1) + g_1(p_1)^2 + e_1(p_1) + g_2(p_5) + e_1(p_2)$	$g_3 + g_3$	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_1) + e_1(p_1) + g_2(p_5) + e_1(p_2)$	s_1
$e_1(p_1) + g_1(p_4) + e_1(p_1) + g_2(p_3) + e_1(p_1)$	g_2	$e_1(p_2) + g_1(p_2)^2 + e_1(p_2) + g_2(p_1) + e_1(p_1)$	$g_3 + g_3$	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_2) + e_1(p_2) + g_2(p_1) + e_1(p_1)$	s_1
$e_1(p_1) + g_1(p_4) + e_1(p_1) + g_2(p_4) + e_1(p_1)$	0	$e_1(p_2) + g_1(p_2)^2 + e_1(p2) + g_2(p2) + e_1(p2)$	g_2	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_2) + e_1(p_2) + g_2(p_2) + e_1(p_2)$	s_1
$e_1(p_1) + g_1(p_4) + e_1(p_1) + g_2(p_5) + e_1(p_1)$	g_1	$e_1(p_2) + g_1(p_2)^2 + e_1(p_2) + g_2(p_3) + e_1(p_2)$	g_1	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_2) + e_1(p_2) + g_2(p_3) + e_1(p_2)$	s_1
$e_1(p_2) + g_1(p_5) + e_1(p_1) + g_2(p_2) + e_1(p_2)$	$g_3 + g_3$	$e_1(p_2) + g_1(p_2)^2 + e_1(p_2) + g_2(p_5) + e_1(p_2)$	g_1^2	$e_1(p_1) + g_1(p_1) + e_1(p_1) + s_1(p_2) + e_1(p_2) + g_2(p_5) + e_1(p_2)$	s_1
$e_1(p_2) + g_1(p_5) + e_1(p_1) + g_2(p_3) + e_1(p_2)$	$g_4 + g_4$	$e_1(p_2) + g_1(p_3)^2 + e_1(p_2) + g_2(p_1) + e_1(p_1)$	g_1^2	$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_1) + g_4(p_4) + e_2(p_2)$	g_4
$e_1(p_2) + g_1(p_5) + e_1(p_1) + g_2(p_4) + e_1(p_1)$	g_2	$e_1(p_2) + g_1(p_3)^2 + e_1(p_2) + g_2(p_2) + e_1(p_2)$	$g_3 + g_3$	$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_2) + g_4(p_5) + e_2(p_3)$	g_4
$e_1(p_2) + g_1(p_5) + e_1(p_1) + g_2(p_5) + e_1(p_2)$	0	$e_1(p_2) + g_1(p_3)^2 + e_1(p_2) + g_2(p_3) + e_1(p_2)$	g_2	$e_1(p_1) + g_1(p_1) + e_1(p_1) + g_2(p_3) + g_4(p_1) + e_2(p_4)$	$g_4 + g_3 + g_3$

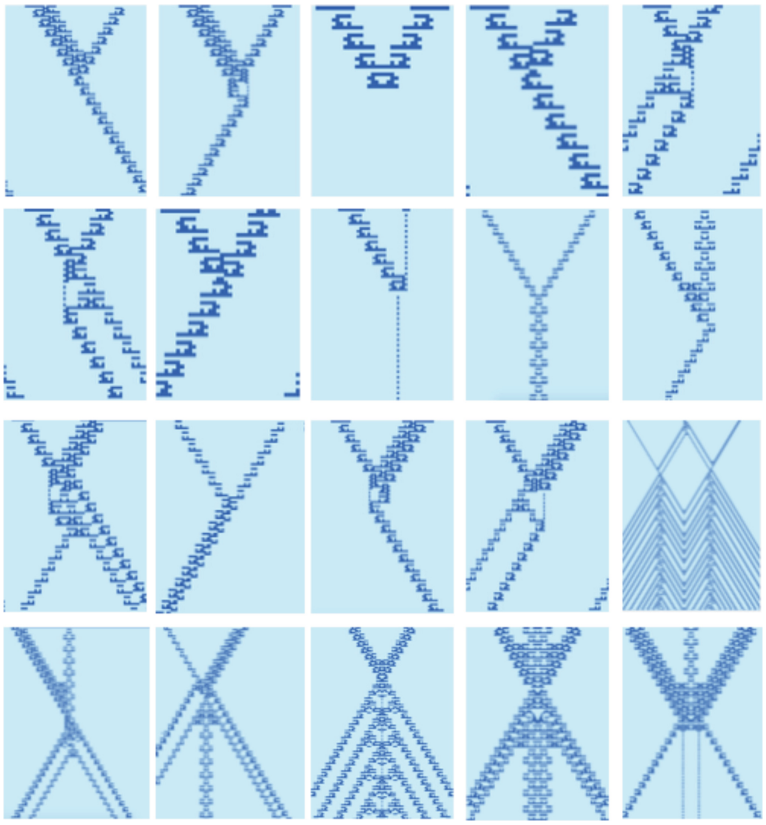

Fig. 4. Codifying gliders in $\phi_{R126m:\tau}$ to controlling collisions. Some examples are illustrated in this figure where we can see annihilations, self-organization, glider production, and glider guns.

5 Conclusions

Memory is a useful tool to discover complexity in dynamical systems from composed functions. In this paper, we have extended these results previously to controller gliders in rule 126 systematically. The next step of our research will design more complex constructions including computable devices by glider collisions.

References

1. Adamatzky, A.: Identification of Cellular Automata. Taylor and Francis, London (1994)
2. Adamatzky, A. (ed.): Collision-Based Computing. Springer, London (2002)
3. Adamatzky, A., Martínez, G.J.: On generative morphological diversity of elementary cellular automata. Kybernetes **39**(1), 72–82 (2010)
4. Aziz-Alaoui, M., Bertelle, C.: From System Complexity to Emergent Properties. Springer, Heidelberg (2009)
5. Bar-Yam, Y.: Dynamics of Complex Systems. Addison-Wesley, New York (1997)

6. Boccara, N.: Modeling Complex Systems. Springer, New York (2003)
7. Chen, B., Chen, F., Martínez, G.J.: Glider collisions in hybrid cellular automaton with memory rule (43,74). Int. J. Bifurc. Chaos **27**(6) (2017). https://doi.org/10.1142/S0218127417500821
8. Chen, G., Dong, X.: From Chaos to Order. In: World Scientific Series on Nonlinear Science, Series A, vol. 24 (1998)
9. Culik II, K., Yu, S.: Undecidability of CA classification schemes. Complex Syst. **2**(2), 177–190 (1988)
10. Kauffman, S.A.: The Origins of Order: Self-organization and Selection in Evolution. Oxford University Press, New York (1993)
11. Martínez, G.J., Adamatzky, A., Sanz, R.A., Mora, J.C.S.T.: Complex dynamic emerging in rule 30 with majority memory a new approach. Complex Syst. **18**(3), 345–365 (2010)
12. Martínez, G.J., Adamatzky, A., Sanz, R.A.: Designing complex dynamics with memory. Int. J. Bifurc. Chaos **23**(10), 1330035–131 (2013)
13. Martínez, G.J., Adamatzky, A., Chen, B., Chen, F., Mora, J.C.S.T.: Simple networks on complex cellular automata: from de Bruijn diagrams to jump-graphs. In: Zelinka, I., Chen, G. (eds.) Swarm Dynamics as a Complex Network, pp. 241–264. Springer, Heidelberg (2018)
14. Martínez, G.J.: A note on elementary cellular automata classification. J. Cell. Autom. **8**(3–4), 233–259 (2013)
15. Martínez, G.J., Adamatzky, A., Mora, J.C.S.T., Sanz, R.A.: How to make dull cellular automata complex by adding memory: rule 126 case study. Complexity **15**(6), 34–49 (2012)
16. Martínez, G.J., Adamatzky, A., Sanz, R.A.: Complex dynamics of cellular automata emerging in chaotic rules. J. Nonlinear Syst. Appl. **22**(2) (2012). https://doi.org/10.1142/S021812741250023X
17. McIntosh, H.V.: Wolfram's class IV and a good life. Physica D **45**, 105–121 (1990)
18. McIntosh, H.V.: One Dimensional Cellular Automata. Luniver Press, Bristol (2009)
19. Mainzer, K., Chua, L.: The Universe as Automaton: From Simplicity and Symmetry to Complexity. Springer, Heidelberg (2012)
20. Minsky, M.: Computation: Finite and Infinite Machines. Prentice Hall, Englewood Cliffs (1967)
21. Mitchell, M.: Complexity: A Guided Tour. Oxford University Press, New York (2009)
22. Martínez, G.J., McIntosh, H.V., Mora, J.C.S.T., Vergara, S.V.C.: Determining a regular language by glider-based structures called phases f_i_1 in rule 110. J. Cell. Autom. **3**(3), 231–270 (2008)
23. Prokopenko, M., Michael Harré, M., Lizier, J.T., Boschetti, F., Peppas, P., Kauffman, S.: Self-referential basis of undecidable dynamics: from The Liar Paradox and The Halting Problem to The Edge of Chaos, CoRR abs/1711.02456 (2017)
24. Sanz, R.A., Martin, M.: Elementary CA with memory. Complex Syst. **14**, 99–126 (2003)
25. Sanz, R.A.: Elementary rules with elementary memory rules: the case of linear rules. J. Cell. Autom. **1**, 71–87 (2006)
26. Sanz, R.A.: Cellular Automata with Memory. Old City Publishing, Philadelphia (2009)
27. Wuensche, A., Lesser, M.: The Global Dynamics of Cellular Automata. Addison-Wesley Publishing Company, Reading (1992)
28. Wolfram, S.: Universality and complexity in cellular automata. Physica D **10**, 1–35 (1984)

29. Wolfram, S.: Cellular Automata and Complexity. Addison-Wesley Publishing Company, Englewood Cliffs (1994)
30. Wolfram, S.: A New Kind of Science. Wolfram Media Inc., Champaign (2002)
31. Wuensche, A.: Classifying cellular automata automatically. Complexity **4**(3), 47–66 (1999)