

Contents

1	Game of Life music	1
	Eduardo R. Miranda and Alexis Kirke	
1.1	A brief introduction to GoL	2
1.2	Rending musical forms from GoL	2
1.3	CAMUS: Cartesian representation of note sets	4
1.3.1	Temporal morphology	5
1.4	CAMUS 3D: Cartesian representation for three dimensional GoL	7
1.5	Radial representation for two dimensional GoL	9
1.6	Concluding remarks	13
	References	14

Chapter 1

Game of Life music

Eduardo R. Miranda and Alexis Kirke

At the time when the first author was post-graduate student, in the evenings he used to entertain himself with the equipment in the electronic music studio at the University of York until dawn. It must have been around three o'clock in the morning of a rather cold winter night in the late 1980s, when he connected his Atari 1040ST computer to a synthesizer to test the first prototype of a system, which he was developing for his thesis. The system, named CAMUS (short for Cellular Automata Music), implemented a method that he invented to render music from the behaviour of the Game of Life (GoL) cellular automata (CA).

It took him a few hours to get the studio and software settings to work. When he finally managed to run the program, the music that is produced took him by surprise: it sounded remarkably interesting! It was an awesome experience, which marked the beginning of his enduring interest in making music with CA. Unwittingly, to the best of our knowledge, Miranda probably was a pioneer of GoL music. He soon learned that two other composers, Peter Beyls (in Belgium) and Dale Millen (in the USA), were also experimenting with CA at the time, but almost certainly not with GoL. Also, it would seem that Iannis Xenakis, a Greek composer then living in Paris, had used CA in the mid of the 1980s “to create complex temporal evolution of orchestral clusters’ for the composition *Horos* ([1]; pp. 122). However, as reported in Hoffman’s paper, this was poorly documented by the composer; it is not clear how the temporal orchestral clusters of *Horos* were generated by the automata.

The animated patterns produced by GoL are rather inspiring and we cannot help but think of ways to render musical forms from the automata rather than — or in addition to — visual patterns.

This chapter introduces the core of Miranda’s original rendering method devised for the CAMUS system and subsequent methods developed in collaboration with Kirke and others.

1.1 A brief introduction to GoL

A detailed introduction to GoL is beyond the scope of this chapter; please refer to the introductory chapters in this volume. Here we present only the very basics necessary to contextualize the chapter.

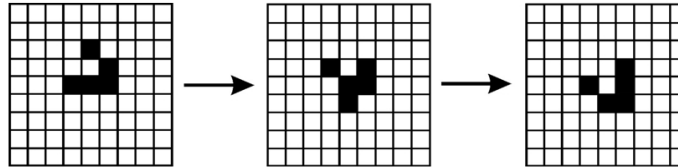


Fig. 1.1 GoL is a two-dimensional cellular automaton where each cell can be in one of two possible states: alive or dead. Given an initial configuration of “living” cells, the application of Conway’s rules results in a sequence of patterns, two of which are shown.

GoL is a two-dimensional CA invented by John Conway where the cells of a matrix of cells can be in one of two possible states: alive or dead (see example of propagating structure in Fig. 1.1). “Conway was fascinated by the way in which a combination of a few simple rules could produce patterns that would expand, change shape, or die out unpredictably. He wanted to find the simplest possible set of rules that would give such an interesting behaviour.” ([5]; pp. 44).

As time progresses, the state of each cell in the matrix is determined by the state of its eight nearest neighbouring cells, as follows:

- Birth: A cell that is dead at time t becomes alive at time $t + 1$ if exactly three of its neighbours are alive at time t
- Death by overcrowding: A cell that is alive at time t will die at time $t + 1$ if four or more of its neighbours are alive at time t
- Death by exposure: A cell that is alive at time t will die at time $t + 1$ if it has one or no live neighbours at time t
- Survival: A cell that is alive at time t will remain alive at time $t + 1$ only if it has either two or three live neighbours at time t

The rules above are applied simultaneously to all cells of the matrix. GoL is also characterised by a number of interesting initial configurations of “living” cells that are known to give rise to intriguing patterning behaviour. A few examples of well-known initial configurations are shown in Fig. 1.2.

1.2 Rending musical forms from GoL

The fundamental challenge of generating music with GoL is to design a suitable method to represent the patterns generated by the automata in terms of musical

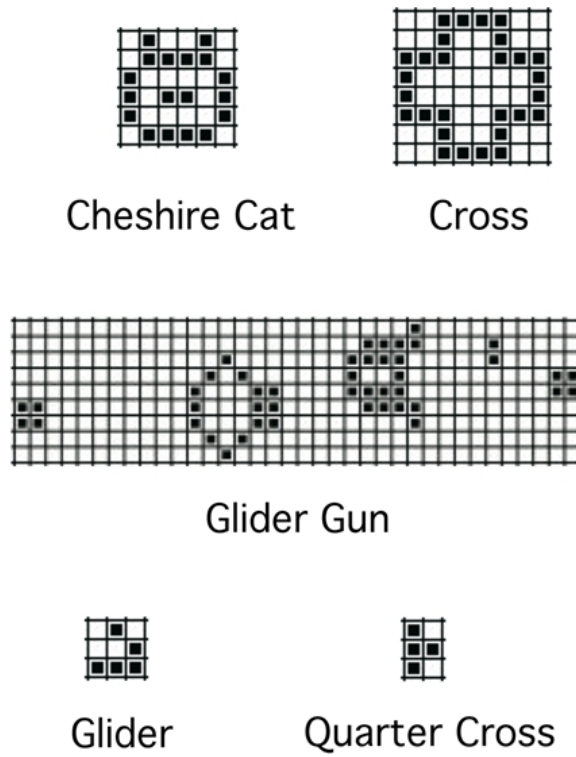


Fig. 1.2 A few examples of initial GL configurations. Note that the “glider” configuration was used in the example in Fig. 1.1.

forms rather than — or in addition to — visual forms. How can one render the sequence of patterns generated by GoL (e.g., the ones shown in Fig. 1.1) in terms of music? In the next three sections we will introduce three mapping approaches for GoL which are summarized in Tab. 1.1.

Name	GoL Dimensions	Coordinate System	Notes per Cell
CAMUS	2	2-D Cartesian	3
CAMUS 3D	3	3-D Cartesian	4
Radial Symmetry	2	2-D Radial	1

Table 1.1 The three types of mappings to be introduced.

1.3 CAMUS: Cartesian representation of note sets

Miranda devised a musical representation scheme, whereby an ordered set of three notes is represented by a point on a two-dimensional Cartesian plane [4]. These three notes are ordered in terms of the distances between them. In musical jargon, the distance between two notes is referred to as an “interval” and the unit is the semitone¹. Given a reference note, the abscissa of the Cartesian plane represents the interval between this reference and the second note of the set. The ordinate represents the interval between the second note and the third (Fig. 1.3).

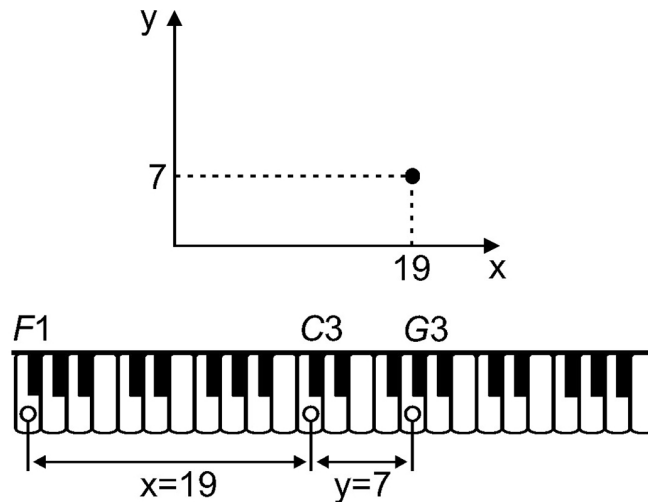


Fig. 1.3 Musical representation using a two-dimensional Cartesian coordinate system to represent an ordered set of three notes. In this case the note C3 is 19 semitones above F1, which is the reference note, and G3 is seven semitones above C3.

Given the representation shown in Fig. 1.3, a GoL matrix of cells can be mapped onto the Cartesian plane, where those cells that are alive become points of the its coordinate system (Fig. 1.4). Thus, as the automaton evolves in time, it generates points representing sets of three notes in terms of coordinates of a two-dimensional Cartesian plane.

To begin a composition process, a $[x \times y]$ GoL automaton is set up with a given initial configuration of “living” cells. At each cycle of the automata (a cycle corresponds to the application of the rules to the cells of the matrix), the cells are analysed

¹ This unit is based on the Western European tempered musical scale consisting of 12 notes separated by a semitone (or half of a tone). For example, on the piano keyboard shown in Fig. 1.3, the distance between C3 and the next white note, D3, is one tone; the distance between C3 and the next back note, C3 sharp, is half of a tone. The seventh white note from C3 is C4; C4 is called an octave of C3. The distance between a note and its octave is 12 semitones. Other cultures use different scales where the interval unit may be different; e.g., quarter of a tone.

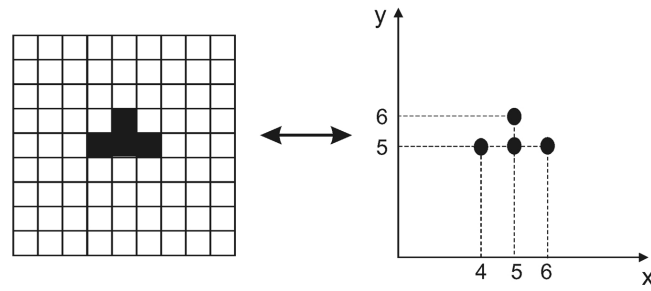


Fig. 1.4 As the automaton evolves, ordered sets of three notes are produced. These sets are defined in terms of the two-dimensional Cartesian coordinate system shown in Fig. 1.3.

column by column, starting with cell $(0,0)$, continuing through to $(0,x)$, moving on to cell $(1,0)$ through to cell $(1,y)$ and continuing in this manner until cell (x,y) has been checked. When the system arrives at a “living” cell, its co-ordinates are used to calculate the three notes, as shown in Fig. 1.3. A set of reference notes is specified beforehand by the user. Although the cell updates occur at each cycle in parallel, the system plays the “living” cells column by column, from top to bottom. The three notes of a set are not necessarily played simultaneously. Rather, they have their own starting and ending times, as explained below.

1.3.1 Temporal morphology

The method for staggering the starting and ending times of the notes of a set represented by a Cartesian point (x,y) uses the states of its respective neighbouring cells in the GoL matrix. The system constructs a set of values from the states of the neighbouring cells, the value being equal to one if the cell is alive and zero if it is dead, as follows:

$$\begin{aligned} a &= \text{cell}(x, y - 1) \\ b &= \text{cell}(x, y + 1) \\ c &= \text{cell}(x + 1, y) \\ d &= \text{cell}(x - 1, y) \\ m &= \text{cell}(x - 1, y - 1) \\ n &= \text{cell}(x + 1, y + 1) \\ o &= \text{cell}(x + 1, y - 1) \\ p &= \text{cell}(x - 1, y + 1) \end{aligned}$$

Then, the system forms four 4-bit words as follows: $abcd$, $dcba$, $mnop$ and $ponm$. Next, it performs the bit-wise inclusive OR operation, ‘—’, to generate two four-bit words: Tgg and Dur :

$$Tgg = abcd|dcba$$

$$Dur = mnop|ponm$$

The system derives trigger information for the notes from Tgg , and duration information from Dur . With each relevant four-bit word, the system associates a code to represent time-forms where B denotes the bottom reference note, M the middle note, and U the upper one. The square brackets are used to indicate that the note events contained within that bracket occur simultaneously. The codes are as follows:

$$\begin{aligned} 0000 &= B[UM] \\ 0001 &= [UMB] \\ 0010 &= BUM \\ 0011 &= UMB \\ 0101 &= BMU \\ 0110 &= UBM \\ 0111 &= MBU \\ 1001 &= U[MB] \\ 1011 &= MUB \\ 1111 &= M[UB] \end{aligned}$$

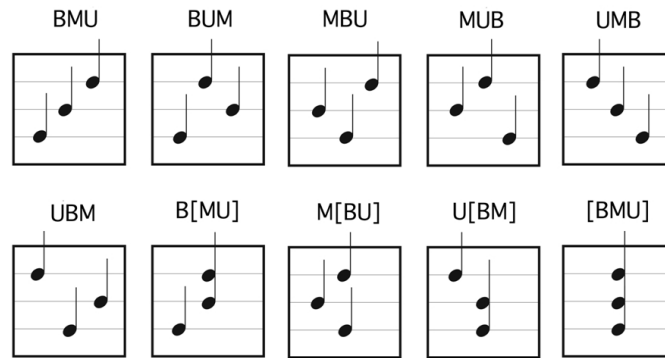


Fig. 1.5 Ten different time-forms combined in pairs define temporal morphologies for the set of three notes associated to a “living” GoL cell.

A visual representation of the time-forms assigned to the 4-bit words are shown in Fig. 1.5. Pairs of time-forms define a temporal morphology for the cells. For example, consider the a temporal morphology starting with MBU and ending with $B[MU]$ (Fig. 1.6). Figure 1.7 shows an instantiation of this morphology in musical notation.

The actual values in milliseconds for the trigger and duration parameters are calculated using a pseudo-random number generator. Finally, the music is written to a MIDI file and/or sent directly to a MIDI sampler or synthesiser to be played. Figure 1.8 illustrates the main steps of the algorithm in the form of a flowchart. An

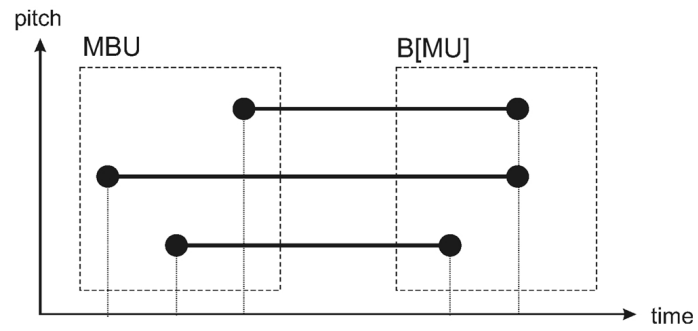


Fig. 1.6 The temporal morphology starting with *MBU* and ending with *B[MU]*.

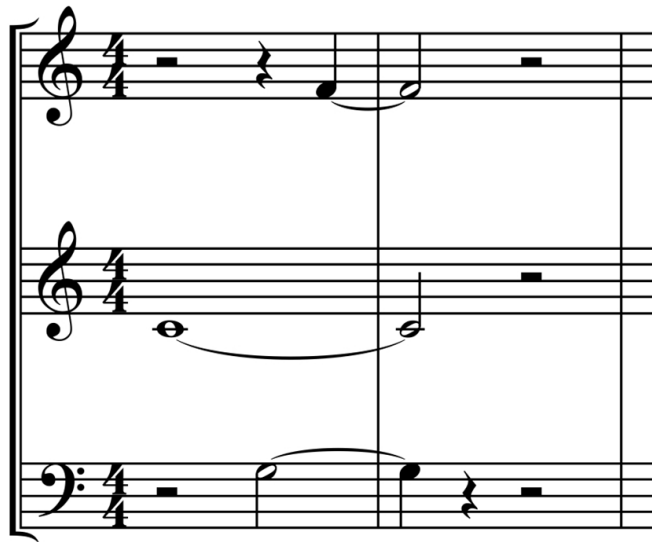


Fig. 1.7 A musical passage generated by a single cell with the temporal morphology portrayed in Fig 1.6.

example of a musical passage generated by the system is shown in Fig. 1.9. Note the characteristic sequence of patterns of groups of three notes evolving in time.

1.4 CAMUS 3D: Cartesian representation for three dimensional GoL

The Cartesian representation described above was extended in collaboration with Kenny McAlpine and Stuart Hoggar, who contributed to further develop CAMUS

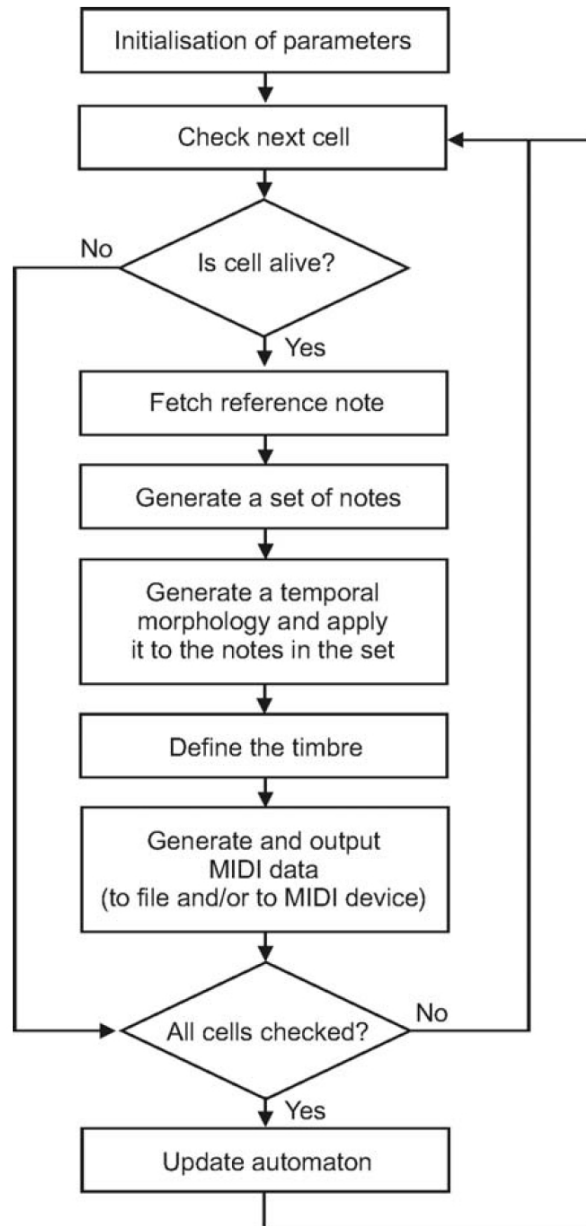


Fig. 1.8 The main steps of the GoL music algorithm.



Fig. 1.9 An example of a musical passage composed with CAMUS. In the first two bars, three patterns of groups of three notes, corresponding to GoL “living” cells are circled.

by introducing a number of variations to the original program, notably the use of three-dimensional GoL [3]. In order to achieve this, the three-dimensional Cartesian space was treated as a series of stacked two-dimensional spaces. Therefore, a three-dimensional GoL was defined as a series of two-dimensional GoL stacked parallel to the plane $x = 0$. Then each of the stacked planes has the form $x = a$, for some integer value, a . Thus, when it comes to assess the neighbouring cells of an arbitrary cell (a, b, c) , the algorithm needs to restrict its attention only to the cells $(a, b + 1, c)$, $(a, b - 1, c)$, $(a, b, c + 1)$, $(a, b, c - 1)$, $(a, b + 1, c + 1)$, $(a, b + 1, c - 1)$, $(a, b - 1, c + 1)$ and $(a, b - 1, c - 1)$, because the focus is on the plane $x = a$. This means that each of the stacked two-dimensional GoL evolve independently; that is, none of the neighbouring cells can exert any influence on the cells in any of the other GoL. This configuration is illustrated in Fig. 1.10.

An example of music generated by the three-dimensional CAMUS is shown in Fig. 1.11. As on the case of Fig. 1.9, it is possible also here to identify the characteristic sequence of patterns of groups of four notes evolving in time.

1.5 Radial representation for two dimensional GoL

In the original Conway’s GoL rules, the state of a cell is determined by the state of the cells around it. If one rotates the neighbourhood of a cell around the centre of the matrix by 90, 180 or 270 degrees, the rule would still have the same effect on the cell’s state. We believe that this is the reason that GoL generates such attractive and musically inspiring symmetric patterns. A number of researchers have discovered what are now well-known symmetrical life evolutions, and there is some discussion of the symmetrical tendencies of GoL [6]. And even for those evolutions, which are non-symmetrical, the emergence of complex broken symmetries from simple symmetrical rules is one element that appeals to our aesthetic sense during GoL iterations.

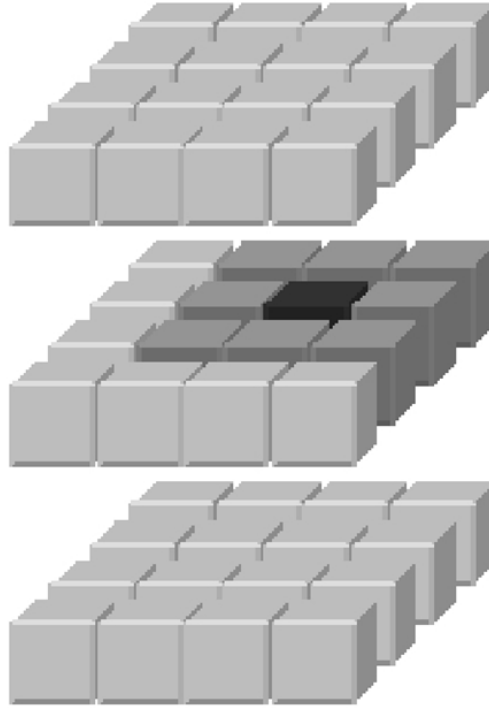


Fig. 1.10 A three-dimensional automaton defined as a series of stacked two-dimensional GoL. The two-dimensional GoL in this case are stacked parallel to the plane $y = 0$. The dark grey cell in the middle layer is currently under examination. Since we treat the y co-ordinate as a constant, only the shaded neighbouring cells in the same plane are also examined. Thus, each two-dimensional game evolves in isolation.



Fig. 1.11 Example of musical passage generated by the three-dimensional CAMUS. In the first two bars, two patterns of groups of four notes, corresponding to GoL “living” cells are circled.

The two mappings introduced so far have focused on a cartesian approach to mapping the cell content. In this third mapping, we devised a method to capture the radial symmetry inherent in the aesthetic of the GoL — polar co-ordinates

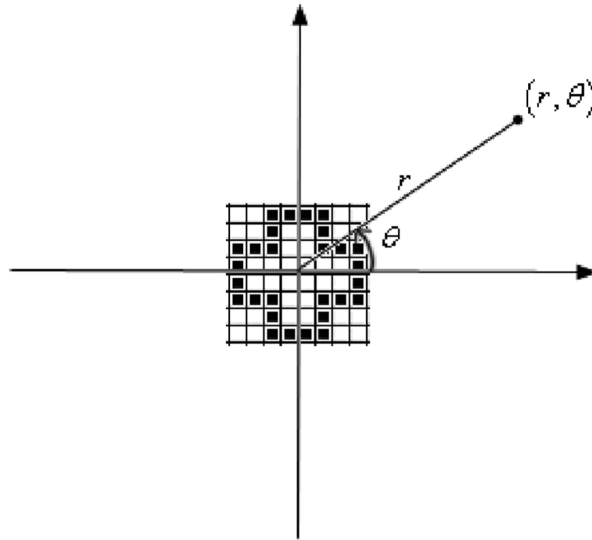


Fig. 1.12 In the polar co-ordinates method to capture the radial symmetry inherent in the aesthetic of the GoL, an origin is defined at the centre of the GoL matrix, and each live cell is mapped into the musical domain based on its (r, θ) co-ordinate.

The following algorithm was developed to generate music using the (r, θ) co-ordinate:

1. Choose a BPM (beats per minute) value. Initialise a variable *baseBeat* to 0, choose a fixed note duration D
2. Run a generation of the GoL
3. Iterate θ around the matrix from 0 to 2π in 127 steps. For each of the 127 values of θ , iterate r from the centre of the matrix to its edge, one cell at a time.
4. For each of the iterated values of r , examine the cell at (r, θ) . If it is “alive” generate a MIDI note with pitch proportional to r , and of duration D . Locate the MIDI note at beat: $baseBeat + D + (16 * (\theta / (2\pi)))$
5. After completing the nested iterations of (r, θ) over the whole matrix, update *baseBeat* to the beat of the last generated MIDI note.
6. Go back to (2) and repeat.

Let us consider the cell with (x, y) co-ordinate $(2, 2)$ in Fig. 1.12. By performing a transformation from Cartesian to Polar co-ordinates we obtain a quantized (r, θ) coordinate of $(3, \pi/2)$. This is halfway across the GoL matrix shown in Fig. 1.12. Thus, the proportional MIDI pitch value will be 64 (halfway up the MIDI pitch

range of 0 to 127). Looking at step (1) above, if we assume a duration D of 1 beat and a starting value of *baseBeat* of 0, then the start of the beat for the MIDI note calculated by step 4 is: $0 + 16(\pi/2)/(2\pi) = 5$. So (2,2) will generate a MIDI note of pitch 64 at beat 5.

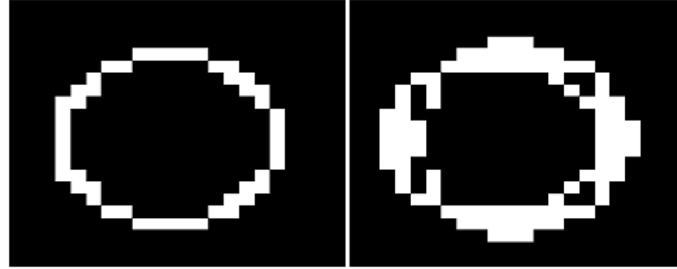


Fig. 1.13 First two generations of a GoL starting with a circle of “alive” cells on the left. (White = “alive’.)

In order to illustrate the radial mapping in practice, let us assume a GoL matrix of size 20 cells by 20 cells is used. The matrix is initialised with a pattern of “living” cells, which is approximately a circle of radius of 7 squares. It is only approximately a circle since it is built using polar co-ordinates, and the grid quantization mapping does not allow it to be a perfect circle. The left hand picture in Fig. 1.13 shows the seed; note: in this case white cells are “alive”, and black ones “dead”. It can be seen that the next cycle of GoL — shown on the right hand side of Fig. 1.13 — does not have full radial symmetry either. However it still has significant amounts of radial symmetry that will be picked up by the generative polar co-ordinate process.

Figure 1.14 shows the resulting output of the two GoL matrices shown in Fig. 1.13. In this example, the pitches were scaled between MIDI values 60 and 90. A tempo of 120 beats per minute was used and the global duration D was set to 2 beats. Looking at the first generation in bars 1 to 8 (corresponding to the left hand side of Fig. 1.13) we can observe the melody falling and then rising again. In fact the melody should stay at a constant pitch for a perfect circle seed. But in reality one side of the “circle” is closer to the origin to the other side due to the generating pattern not being a perfect circle around the origin; that is, a quantization error for such a small circle in GoL. Furthermore, in bars 1 to 8, there would only be a single pitch at any time if we had a perfect circle. But the way the circle is generated in the seed causes it to have a thickness of 2 cells in certain places. From bar 9 onwards the second generation (on the right hand side of Fig. 1.13) is playing. Some of the “rise, fall, rise”-effect in pitch, seen in the first 8 bars, is still there. This is because the second generation still approximately holds the quantized circle shape. However, as can be seen in the right hand side of Fig. 1.13, the complexity of GoL is now starting to manifest in the score. A significant number of new cells have been brought to life, contributing to the harmony of the music. Furthermore, the quantization of the circle leads to θ generating slightly more complex rhythms.



Fig. 1.14 Music of the first 2 generations of the GoL shown in Fig. 1.13.

Despite the quantization errors, these two generations show how the polar mapping is starting to capture the outward symmetrical complex movement of the circle seed. Thus better capturing some of the radially symmetric aesthetic of the visuals than a linear mapping would.

1.6 Concluding remarks

Pioneers such as Xenakis, Beyls, Millen and others, rightly understood the suitability of cellular automata to model systems that change some feature with time. Essentially, music is a time-based art form, where sequences of musical notes and rhythms form patterns of sonic structures organised in time. In this context, GoL is appealing because they produce sequences of coherent patterns, some of which can be very complex, and yet controlled by remarkably simple rules. In this chapter we have introduced three mapping approaches to composing music with GoL — a two-dimensional and three-dimensional Cartesian mapping, and a two-dimensional radial mapping. By no means do these exhaust the mapping possibilities, but they highlight some key aspects of GoL musical mappings, as shown earlier in Tab. 1.1.

The systems described in this chapter have been used to compose a number of fully-fledged pieces of music professionally. In relation to this, it should be noted that an inevitable problem with the great majority of systems that generate music automatically is that they do not possess knowledge about musical instruments. They often generate musical passages that would be technically impossible to be played

on the respective instruments. These pieces tend to sound unconvincing, or “non-musical”, when played on a MIDI synthesiser or sampler, because the music is not performed idiomatically; the clarinets do not sound “clarinetistically”, the violins do not sound “violinistically”, and so on. Better results can be achieved by amending the score manually in order to render the piece more realistic. The systems described in this chapter are not exceptions. For instance, the passages in Fig. 1.9 have been edited manually in order to alleviate the idiomatic problem. However, the edits in Fig. 1.9 were kept to a minimum and often boiled down to transposition of a note an octave upwards or downwards. Nevertheless, it is still possible to identify the characteristic sequence of patterns of groups of three notes evolving in time. Dynamics and articulation were also added manually.

The resulting pieces were performed in public concerts and some are available on CD. For instance, there are two compositions on the CD *Plural*, released in Brazil (389.726.04-7): tracks 1-3 contains the piece *Grain Streams* for piano and electronic sounds and track 14 contains the piece *Entre of Absurdo e o Mistério*, for orchestra. Please e-mail the authors should you wish to order a copy of this CD or obtaining recordings of other GoL-generated pieces.

References

1. Hoffman, P. (2002). Towards and Automated Art: Algorithmic Processes in Xenakis' Compositions, *Contemporary Music Review* 21(2-3):121-131.
2. Kirke, A., Miranda, E.R. (2007). Capturing the Aesthetic: Radial Mappings for Cellular Automata Music. *Journal of the ITC Sangeet Research Academy* 21:15-23.
3. McAlpine, K. Miranda, E. R. and Hoggar, S. (1999). Making Music with Algorithms: A Case Study System, *Computer Music Journal* 23(2):19-30.
4. Miranda, E. R. (1993). Cellular Automata Music: An Interdisciplinary Project, *Interface* 22(1):3-21.
5. Wilson, G. (1988). The life and times of cellular automata, *News Scientist* October 1988:44-47.
6. Wolfram, S. (1994). Universality and Complexity in Cellular Automata, *Physica D* 10:1-35.