Chapter

Memristive Grid for Maze Solving

Arturo Sarmiento-Reyes and Yojanes Rodríguez Velásquez

Abstract

Memcomputing represents a novel form of neuro-oriented signal processing that uses the memristor as a key element. In this chapter, a memristive grid is developed in order to achieve the specific task of solving mazes. This is done by resorting to the dynamic behavior of the memristance in order to find the shortest path that determines trajectory from entrance to exit. The structure of the maze is mapped onto the memristive grid, which is formed by memristors that are defined by fully analytical charge-controlled functions. The dependance on the electric charge permits to analyze the variation of the branch memristance of the grid as a function of time. As a result of the dynamic behavior of the developed memristor model, the shortest path is formed by those memristive branches exhibiting the fastest memristance change. Special attention is given to achieve a realistic implementation of the fuses of the grid, which are formed by an anti-series connection of memristors and CMOS circuitry. HSPICE is used in combination with MATLAB to establish the simulation flow of the memristive grid. Besides, the memristor model is recast in VERILOG-A, a high-level hardware description language for analog circuits.

Keywords: memristive grids, symbolic memristor modeling, maze-solving, analog processors

1. Introduction

For thousands of years, mazes have intrigued the human mind [1]. The labyrinths have been used in research with laboratory animals, in order to study their ability to recognize their environment [2–4]. In the 1990s, artificial intelligence of robots was studied by examining their ability to traverse unfamiliar mazes [5–7]. Maze exploration algorithms are closely related to graph theory and have been used in both mathematics and computer science [8, 9].

There are several algorithms for maze solving in the literature, they can be classified in two very well-defined groups: the algorithms used by a traveler in the maze without knowledge of a general view of the maze, and the algorithms used for a program that can have a whole view the whole maze. Some examples of the first ones are the wall follower, random mouse, pledge algorithm [10], and Trémaux's algorithm [11]. In the second group, shortest path algorithms are most useful, because they can find the solution not only for a simple connected maze, but also for multiple-solution mazes.

In this chapter, we put a main idea into practice, namely that the topology of a maze can be mapped onto a memristive grid. By exploiting the analog computations performed by solving Kirchoft's Current Laws (KCL) in a parallel manner, memristive grids have demonstrated their ability for computing shortest paths in a

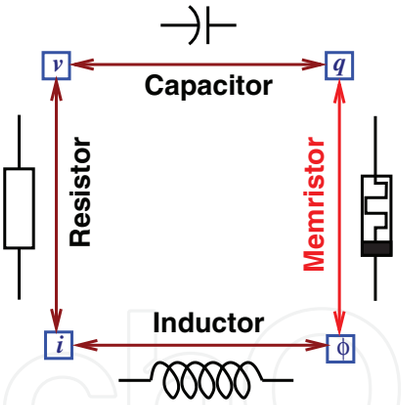

Figure 1.
Basic circuit elements.

given maze, levering on the dynamic adjustment of their intrinsic memristance [12, 13].

The parallel solution of KCL introduces a resemblance of the memristive grid as an analog processor [14] in counterposition to a digital approach in which the processing can also be done in parallel way, but the overhead in additional conversion circuitry is too high.

Two important milestones appear in the history of the memristor. The first one in 1971 when professor Leon O. Chua introduced the memristor as the fourth basic circuit element in his seminal paper [15]. It established that the memristor completes the number of possible relationships between the four fundamental circuit variables: current, voltage, magnetic flux, and electric charge—as depicted in **Figure 1**. Later, an extension to memristive systems was published in [16].

The second milestone occurred in 2008, when a team at Hewlett-Packard Laboratories fabricated a device whose behavior exhibited the memristance phenomenon [17]. Since the advent of the memristor as an actual device, research and technological development in several areas related to memristive applications have been increased.

In the field of signal processing, the memristor has special preponderance in neuro-computing and artificial neural networks because it allows new architectures and processing paradigms with important features based on biological neuronal systems [18–22]. In summary, a novel form of neuro-computing is on scene, namely memcomputing [23].

Memristive grids represent a family of neuro-computing systems that are able of achieving in a very flexible way several tasks for analog applications. In the next paragraphs, we present a specially tailored memristive grid that is focused on solving mazes.

The rest of the manuscript is organized as follows: in Section 2, the developed models are recast in a set of fully analytical expressions for the memristance, which are given as charge-controlled functions that are further used in this application. The components of the memristive grid are introduced in Section 3. The maze-solving procedure is introduced in Section 4 by explaining the simulation flow of the memristive grid. Subsequently, several mazes are solved in order to illustrate the operation of the memristive grid in Section 6. Finally, a series of conclusions is drawn.

2. Development of a charge-controlled memristor model

In this section, a charge-controlled memristor model is introduced. The model has been developed by solving the ordinary differential equation (ODE) that

describes the nonlinear drift mechanism, with a homotopy perturbation method that yields an analytical expression for the memristance [24–27].

In order to obtain a charge-dependent memristance model, the nonlinear drift differential equation is expressed in terms of the electric charge:

$$\frac{dx(q)}{dq} = \eta \kappa f_w(x(q)) \tag{1}$$

where η defines the direction of the drift and it can be ± 1. Besides, f_w is the window function used to define the nonlinear and bounded behavior of the state variable $x(q)$, and it is given as [28]:

$$f_w = 1 - (2x - 1)^{2k} \tag{2}$$

Figure 2 shows the resulting window plots for various values of k. In addition, κ is given as:

$$\kappa = \frac{\mu R_{on}}{\Delta^2} \tag{3}$$

where μ, R_{on}, and Δ are the mobility, the ON-state resistance, and the dimension of the device.

The main goal is to obtain a solution to Eq. (1) in the form of an analytical expression x(q). Once, this is done, this solution is substituted into the coupled resistor equivalent of **Figure 3** which is expressed as [17]:

$$M(t) = R_{on}x(q) + R_{off}[1 - x(q)] \tag{4}$$

where $M(t)$ is the total memristance. Besides, R_{on} and R_{off} are the on-state and the off-state resistances respectively.

In order to obtain an analytical solution to Eq. (1), we resort to the methodology reported in [24, 29], which is based on the homotopy perturbation method (HPM). HPM finds $x(q)$ for a given order of the homotopy method as well as the integer value of exponent of the window function (k). Furthermore, it should be also pointed out that the charge may be positive or negative.

As a result, the sign of the charge as well as the direction of the drift (η) allows us to introduce two operators that are used to simplify the final expressions for the solution. These operators are denoted as Λ and Θ. **Table 1** shows how they are defined depending on the signs of the charge and η.

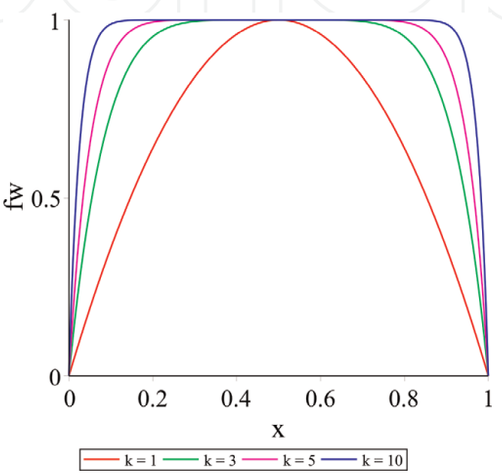

Figure 2.
Window function for different values of k.

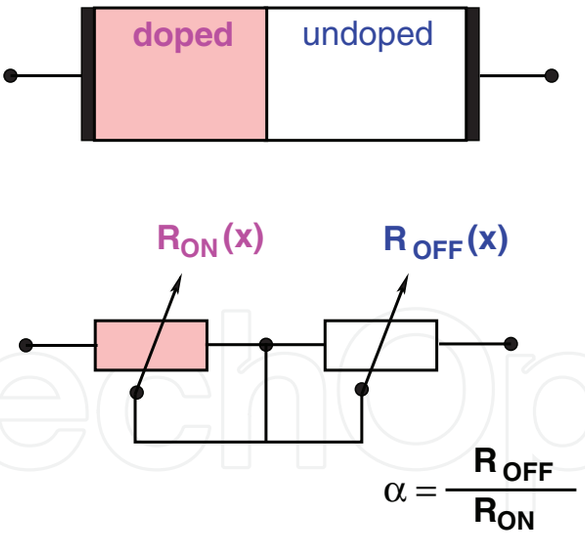

Figure 3.
Coupled series equivalent of the memristor.

	$q \geq 0$	$q < 0$
η^+	$\Lambda = -1$ $\Theta = 1$	$\Lambda = 1$ $\Theta = 0$
η^-	$\Lambda = -1$ $\Theta = 0$	$\Lambda = 1$ $\Theta = 1$

Table 1.
Operators for the signs of η and q.

As a matter of an example, the expression of $x(q)$ for order-1 with $k = 1$ is given as:

$$X_{O1K1}(q) = \Theta\left[1 + (X_0 - 1)^2 e^{8\Lambda\kappa q} - (X_0 - 1)(X_0 - 2)e^{4\Lambda\kappa q}\right] + (1 - \Theta)X_0\left[-X_0 e^{8\Lambda\kappa q} + (X_0 + 1)e^{4\Lambda\kappa q}\right] \tag{5}$$

After substituting Eq. (5) in Eq. (4), it results in the memristance expression:

$$\begin{aligned} M_{O1K1} = {} & \Theta\left[R_d(Xo - 1)\left[(X_0 - 2)e^{\Lambda 4\kappa q} - (Xo - 1)e^{\Lambda 8\kappa q}\right] + R_{ON}\right] \\ & + (1 - \Theta)\left[R_d X_0\left[X_0 e^{\Lambda 8\kappa q} - (X_0 + 1)e^{\Lambda 4\kappa q}\right] + R_{off}\right] \end{aligned} \tag{6}$$

where the variable R_d is given as:

$$R_d = R_{off} - R_{on} \tag{7}$$

For order-2 and $k = 1$, the solution to Eq. (1) is given as:

$$\begin{aligned} X_{O2K1}(q) = {} & \Theta\left[1 + (X_0 - 1)\left(X_0^2 - 3X_0 + 3\right)e^{4\Lambda\kappa q} - (X_0 - 1)^2(2X_0 - 3)e^{8\Lambda\kappa q} + (X_0 - 1)^3 e^{12\Lambda\kappa q}\right] \\ & + (1 - \Theta)\left[X_0\left(X_0^2 + X_0 + 1\right)e^{4\Lambda\kappa q} - X_0^2(2X_0 + 1)e^{8\Lambda\kappa q} + X_0^3 e^{12\Lambda\kappa q}\right] \end{aligned} \tag{8}$$

Again, after substituting the expression above in Eq. (4) and after some reductions, it is possible to obtain the memristance for order-2 and $k = 1$ as:

$$\begin{aligned} M_{O2K1} = {} & M_{O1K1} + Rd\left[\Theta(Xo - 1)^3(-e^{\Lambda 4\kappa q} - 2e^{\Lambda 8\kappa q} - e^{\Lambda 12\kappa q})\right. \\ & \left. + X_0^3(-e^{\Lambda 12\kappa q} - 2e^{\Lambda 8\kappa q} - e^{\Lambda 4\kappa q})(1 - \Theta)\right] \end{aligned} \tag{9}$$

In a similar way, an expression for the memristance for order-3 and $k = 1$ can be obtained:

$$\begin{aligned} M_{O3K1} = M_{O2K1} + Rd\Big[&\Theta(Xo-1)^4\left(-e^{\Lambda 4\kappa q} - 3e^{\Lambda 8\kappa q} - 3e^{\Lambda 12\kappa q} - e^{\Lambda 16\kappa q}\right) \\ &+X_0^4\left(-e^{\Lambda 16\kappa q} - 3e^{\Lambda 12\kappa q} - 3e^{\Lambda 8\kappa q} - e^{\Lambda 4\kappa q}\right)(1-\Theta)\Big] \end{aligned} \tag{10}$$

It can be noticed that HPM produces nested expressions of the memristance, that is to say, a given memristance of a given order is expressed as function of the memristance of lower orders.

For order-1 and $k = 2$, the memristance is given as follows:

$$\begin{aligned} M_{O1K2} = R_d\Theta&\left[\begin{array}{c} \frac{4}{3}\left(X_0^2+1\right)\left(X_0^2-2X_0+2\right)e^{8\Lambda\kappa q} - 3\left(2X_0^2-2X_0+1\right)e^{16\Lambda\kappa q} \\ +2\left(3X_0^2-3X_0+1\right)e^{24\Lambda\kappa q} - \left(\frac{4}{3}X_0^4 - \frac{8}{3}X_0^3 + 4X_0^2 - \frac{8}{3}X_0 + \frac{2}{3}\right)e^{32\Lambda\kappa q} - 1 \end{array}\right] \\ &+ R_d\left[\begin{array}{c} -\frac{1}{3}X_0\left(2X_0^3 - 6X_0^2 + 9X_0 + 3\right)e^{8\Lambda\kappa q} \\ +3X_0^2e^{16\Lambda\kappa q} - 2X_0^3e^{24\Lambda\kappa q} + \frac{2}{3}X_0^4e^{32\Lambda\kappa q} \end{array}\right] + R_{off} \end{aligned} \tag{11}$$

In a similar way, the memristance for order-2 and $k = 2$ is given:

$$\begin{aligned} M_{O2K2} = M_{O1K2} + R_d&\left[\begin{array}{c} \Theta\left(\begin{array}{c} \mathcal{P}_1e^{8\Lambda\kappa q} + \mathcal{P}_2e^{16\Lambda\kappa q} + \mathcal{P}_3e^{24\Lambda\kappa q} + \mathcal{P}_4e^{32\Lambda\kappa q} \\ +\mathcal{P}_5e^{40\Lambda\kappa q} + \mathcal{P}_6e^{48\Lambda\kappa q} + \mathcal{P}_7e^{56\Lambda\kappa q} \end{array}\right) \\ -\frac{1}{45}X_0^3\mathcal{P}_8e^{8\Lambda\kappa q} + 2X_0^3\mathcal{P}_9e^{16\Lambda\kappa q} - X_0^3\mathcal{P}_{10}e^{24\Lambda\kappa q} \\ +\frac{8}{9}X_0^4\mathcal{P}_{11}e^{32\Lambda\kappa q} - 13X_0^5e^{40\Lambda\kappa q} + \frac{24}{5}e^{48\Lambda\kappa q} - \frac{8}{9}e^{56\Lambda\kappa q} \end{array}\right] \end{aligned}$$

$$\begin{aligned} \mathcal{P}_1 &= -\frac{128}{45}X_0^6 + \frac{128}{15}X_0^5 - \frac{89}{9}X_0^4 + \frac{50}{9}X_0^3 + \frac{11}{3}X_0^2 - \frac{226}{45}X_0 + \frac{106}{45} \\ \mathcal{P}_2 &= 4X_0^4 - 8X_0^3 - 22X_0^2 + 26X_0 - 10 \\ \mathcal{P}_3 &= 8X_0^6 - 24X_0^5 + 36X_0^4 - 32X_0^3 + 75X_0^2 - 63X_0 + 19 \\ \mathcal{P}_4 &= -\frac{16}{9}X_0^6 + \frac{16}{3}X_0^5 - \frac{488}{9}X_0^4 + \frac{896}{9}X_0^3 - \frac{400}{3}X_0^2 + \frac{760}{9}X_0 - \frac{184}{9} \\ \mathcal{P}_5 &= 65X_0^4 - 130X_0^3 + 130X_0^2 - 65X_0 + 13 \\ \mathcal{P}_6 &= \left(2X_0^2 - 2X_0 + 1\right)\left(X_0^4 - 2X_0^3 + 5X_0^2 - 4X_0 + 1\right) \\ \mathcal{P}_7 &= \frac{56}{9}X_0^6 - \frac{56}{3}X_0^5 + \frac{280}{9}X_0^4 - \frac{280}{9}X_0^3 + \frac{56}{3}X_0^2 - \frac{56}{9}X_0 + \frac{8}{9} \\ \mathcal{P}_8 &= 40X_0^4 - 204X_0^3 + 495X_0^2 - 630X_0 + 405 \\ \mathcal{P}_9 &= 2X_0^2 - 6X_0 + 9 \\ \mathcal{P}_{10} &= 4X_0^3 - 12X_0^2 + 18X_0 + 9 \\ \mathcal{P}_{11} &= 2X_0^3 - 6X_0^2 + 9X_0 + 18 \end{aligned} \tag{12}$$

Eqs. (6), (9)–(12) are indeed the analytical expressions that constitute memristor models. References [29, 30] contain a proper characterization of the resulting models.

3. Implementing the memristive grid

A memristive grid is a rectangular array of memristive branches, as shown in **Figure 4**. Herein, the memristive branches have been denoted as *bricked* circuit elements called memristive fuses. In addition, a memristive fuse is composed of a series connection of two memristors in anti-series and a switching device [14].

The switch is used to define the structure of the labyrinth, if the switch is in the ON-state, then the way is free, while if the switch is in the OFF-state then a wall is encountered. **Figure 5** shows the equivalent of the memristive fuse.

In order to illustrate the use of the memristive grid in describing a maze, the maze of **Figure 6a** is used. The entrance of the maze is marked by the green arrow and the output is marked by a red arrow, and the walls are shown in red. The maze is mapped onto the memristive grid as shown in **Figure 6b** by denoting the entrance of the maze as a voltage source, while the output of the maze is given by the ground node. For sake of clarity, both figures are merged into **Figure 6c**, where

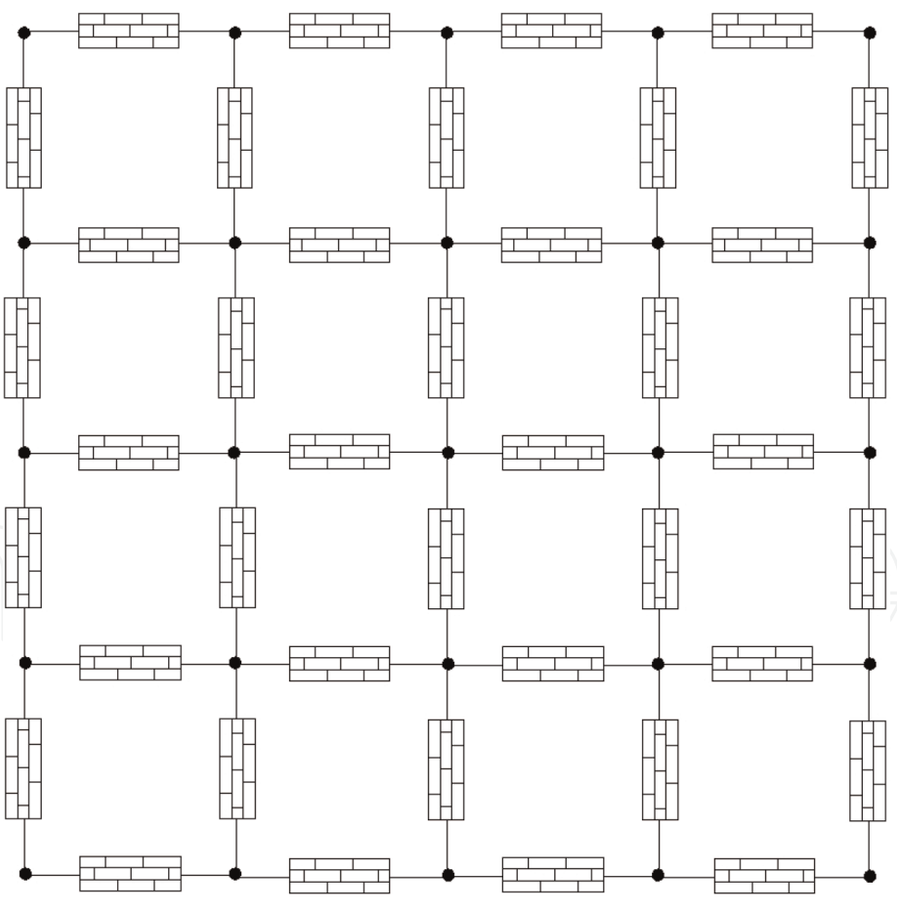

Figure 4.
Description of the memristive grid.

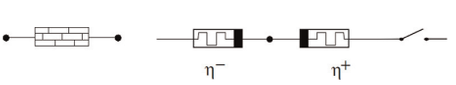

Figure 5.
Configuration of the memristive fuse for maze solving.

the blocked paths are represented by memristive fuses in red, on the contrary, the paths that can be followed are represented by memristive fuses in white. It clearly results that the walls should be given by memristive fuses with the switch in the OFF-state (high-resistance), while the open paths are constituted by memristive fuses with the switch in the ON-state (low-resistance).

On the top of this, the memristive grid can be straightforwardly adapted to other kinds of mazes. Mazes with multiple entrances are represented with multiple input voltages. Similarly, mazes with multiple outputs are given by setting multiple instances of the ground node.

3.1 An algorithmic view

A close look of the solution path in **Figure 6a** can lead us to a graph-theoretical explanation on how the memristive grid solves the maze, because the open ways in the maze can be regarded as an unweighted graph where the solution path is subgraph. The solution path can be found by using a *breath-first-search* (BFS) algorithm in order to traverse the graph which yields indeed the shortest-path because we deal with an unweighted graph [31].

The application of BFS is illustrated by determining the shortest path between nodes 3 and 6 of the graph from **Figure 7a**. Here, node 3 can be regarded as the input (i) and node 6 as the output (o). The algorithm starts by selecting the initial node (3). From this, a first level of coloring is achieved by selecting the neighboring nodes (2, 4, 5). This procedure is repeated until all nodes have been visited. For this graph, it suffices with 2 levels. The shortest path is defined by the sequence 3→5→6, which is shown in red in **Figure 7b**.

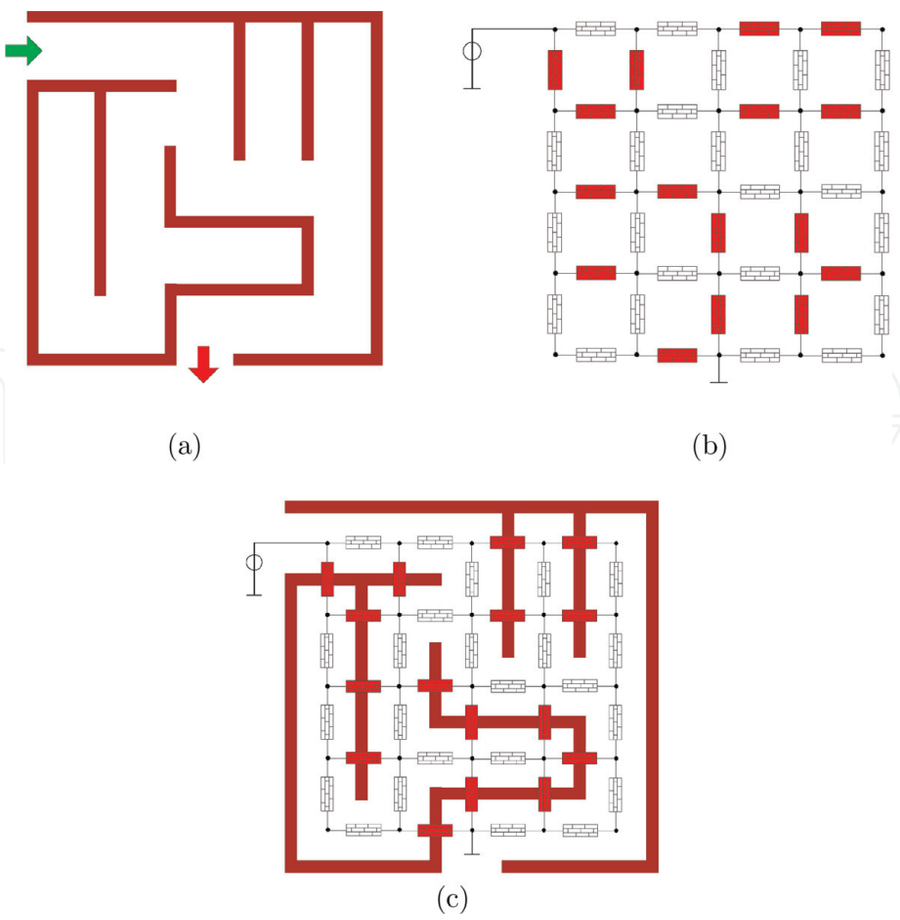

Figure 6.
Mapping the maze onto the memristive grid. (a) Maze, (b) Grid and (c) Merging the maze and the grid.

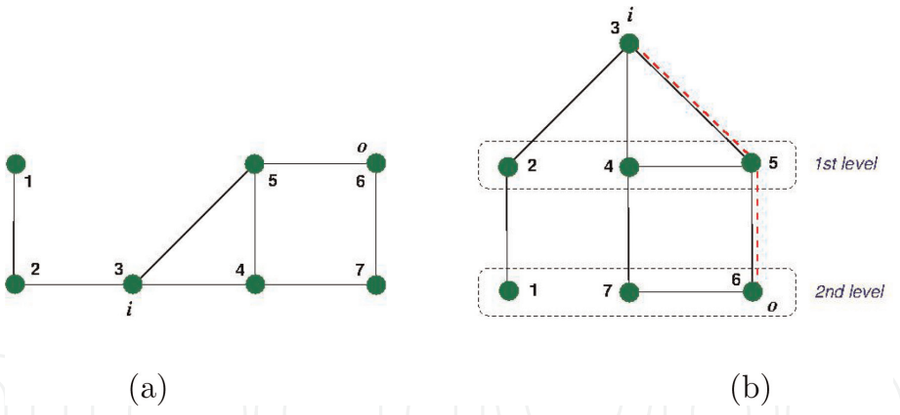

Figure 7.
BFS algorithm to obtain the shortest path. (a) A graph and (b) The BFS algorithm.

As a result, by representing the graph with the memristive grid, it allows us to define ways for the current to flow through the open paths by gradually changing the equivalent memristance of the fuse. Besides, what is more relevant, since the current is given as the time-derivative of the charge, then the solution of the maze is always given by the shortest path to ground which represents the path with the fastest changing memristance.

3.2 Technical specifications of the memristive fuse

The memristive fuse from **Figure 5** contains a pair of memristors in anti-series connection. Such a memristor connection produces an M-q characteristic that is composed of the overlapping of the M-q curves of the memristor expressions for η^- and η^+. **Figure 8a** shows the M-q characteristics for the model of order-1, $k = 5$ and **Figure 8b** shows the schematic curve with the values of R_{off} and R_{init}. Physical parameters of the memristor model are given by the nominal values of the HP memristor. A summary of the specs for the memristor model is given in **Table 2**.

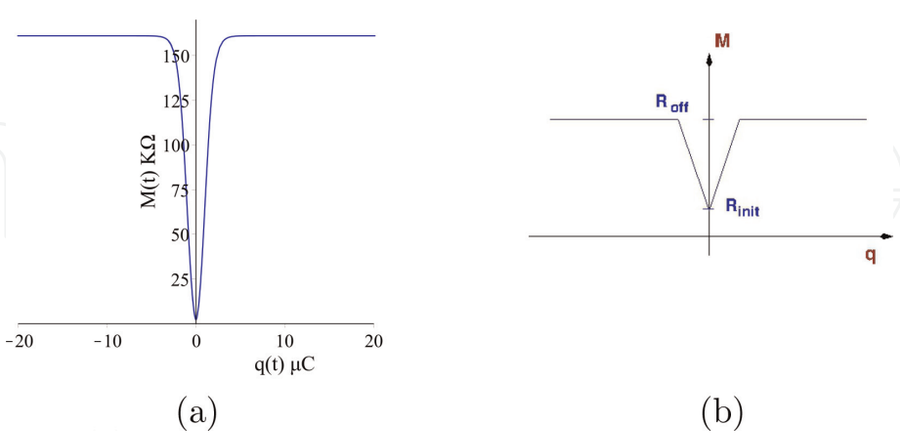

Figure 8.
Memristance-charge characteristic of the anti-series connection. (a) M_{O1K5} and (b) M-q.

$\mu_v \left[\frac{m^2}{Vs}\right]$	Δ $[nm]$	R_{on} $[\Omega]$	R_{off} $[\Omega]$	R_{init} $[\Omega]$	k	Order
1×10^{-14}	10	100	16×10^3	1×10^3	5	1

Table 2.
Memristor parameters of the anti-series connection.

It clearly results that the overall performance of the grid in solving mazes is based on the model of the memristors that form the fuses. Even though the models are recast in fully symbolic form—which represent a great advantage, numeric values should be assigned to the parameters of the model, as given in **Table 2**. Since variations of the model parameters may appear, it is important to notice that the anti-series connection alleviates the possible effects of those variations. Specific sensitivity analysis on the parameter variations of the charge-controlled models are given in [30].

3.2.1 Switch implementation

In the memristive fuse, an ideal switch can be used in the process of finding the solution, however, with the aim to have a more realistic switch, a transmission gate is used instead. The transmission gate is a switch in CMOS technology, it consists of an NMOS transistor and a PMOS transistor connected in parallel, as in **Figure 9a**. Both devices in combination can fully transmit any signal value between V_{dd} (the supply voltage of the transistors) and ground. In order to switch, each transistor requires a complementary control input. Therefore, it is necessary to add an inverter connected between the control input and the PMOS gate [30, 32].

If the control input is V_{dd} then the switch is closed, and as a result, the transmission gate can pass the input signal to output because it exhibits a low-resistance. On the contrary, if the control input is grounded, then the switch is opened and the transmission gate presents a high-resistance.

In order to simulate the transmission gate of the memristive fuse, a CMOS 180 nm technology is used. The parameters of the two complementary transistors are shown in **Table 3**. The equivalent resistance of the transmission gate both states as a function of the input voltage is shown in **Figure 10**.

The resistance values are extracted making a sweep of the input voltage and measure the equivalent average resistance of the transistors in the ON-state

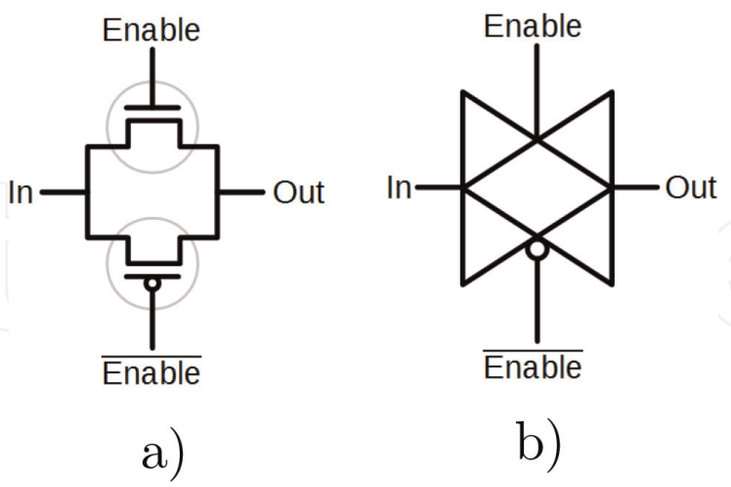

Figure 9.
Transmission gate. (a) Configuration and (b) symbol.

CMOS TG	W μm	L μm
PMOS	1.44	0.18
NMOS	0.48	0.18

Table 3.
Transmission gate: transistor parameters.

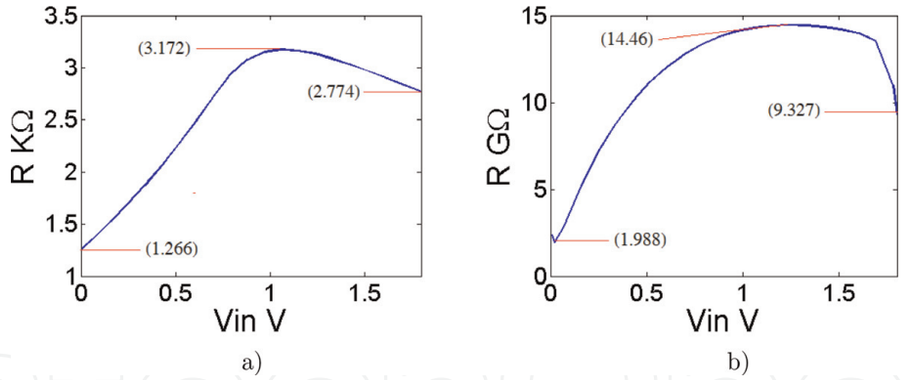

Figure 10.
Resistance characteristic of the transmission gate for both states. (a) ON-state and (b) OFF-state.

$R_{TG_{on}}$ Ω	$R_{TG_{off}}$ Ω
2.504×10^3	10.854×10^9

Table 4.
Selected values for $R_{TG_{on}}$ and $R_{TG_{off}}$.

(switch closed, **Figure 10a**) and OFF-state (switch opened, **Figure 10b**). **Table 4** shows the selected values for $R_{TG_{on}}$ and $R_{TG_{off}}$.

In addition, it can be noticed that the initial value of the ON-state resistance is given as:

$$R_{TG_{init}} = R_{TG_{on}}|_{Vin=0} = 1.266 \tag{13}$$

As a result of the specifications above, a couple of parameters are of special interest, namely, the initial resistance and the maximum resistance of the memristive fuses. At the start, the fuses present an initial resistance which is given as the sum of the initial resistance of the memristors in the anti-series connection plus initial resistance of the ON-state of the transmission gate:

$$R_{fuse_{init}} = 2R_{init} + R_{TG_{init}} \tag{14}$$

which is 3.266 kΩ.
Moreover, the maximum resistance of the fuse is given as:

$$R_{fuse_{max}} = R_{off_1} + R_{on_2} + R_{TG_{on}} \tag{15}$$

It is worthy to notice that the maximum fuse resistance does not contain R_{off} of both memristor, but R_{off} of one memristor and R_{on} of the other memristor due to the anti-series connection.

4. Simulation flow

Since the solution path for a given maze is obtained by determining the path where the fastest change in resistance occurs, the core of the solution process involves a transient analysis. We have chosen to achieve the electrical simulation of the memristive grid by using HSPICE. Both memristors of the fuse are defined as nonlinear resistors in the input netlist.

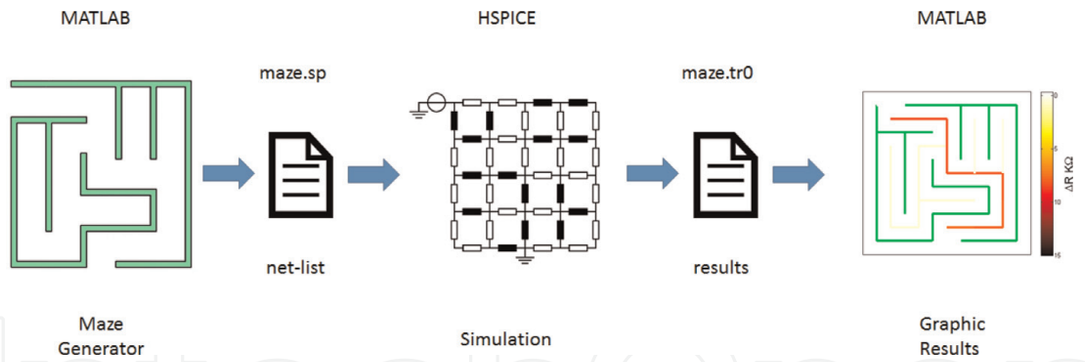

Figure 11.
Simulation flow.

The simulation flow is shown in **Figure 11** and is described as follows:

Maze generator: The first stage in the solution process is to generate the maze by using a script in MATLAB that generates the maze and it is shown as a plot. The walls of the maze are shown in green color in the resulting plot. From this graphical description, the maze can be automatically mapped onto the memristive grid and an input file for HSPICE is generated. The inputs in the maze are represented by input voltage sources of 1 V and the exits are connected to ground.

Electric simulation: The netlist obtained by the maze generator is simulated with HSPICE. Here, a transient analysis for 20 s is carried out, this time is enough to find the solutions of the mazes under-test, however, the exact time when the solutions are found depends on the maze dimensions (grid). The transient simulation results are saved in a .tr0 output file.

Graphic display of the results: In order to visualize the results, a script in MATLAB imports the output simulation signals obtained with HSPICE. The resistance dynamic change ($\Delta R(t)$) is calculated at each simulation time and then the paths of the maze are represented by a graph, where the color in each branch indicates the level of $\Delta R(t)$ at a given time. For sake of readiness, we have selected white for the minimum change and black for the maximum change.

During the transient simulation, the equivalent resistance of the fuses is obtained at every instant t. It clearly results that ΔR is obtained by calculating the difference between the measured resistance and the minimum resistance from Eq. (14):

$$\Delta R(t) = R(t) - R_{fuse_{init}} \tag{16}$$

Consequently, the fuses that first reach the highest ΔR define indeed the solution path of the maze. In mazes with multiple solutions, fuses that belong to the shortest path reach high values of ΔR more fastly. As time lapses, other solution paths are revealed reaching high values of ΔR. For a given time, all fuses within the solution paths reach the maximum ΔR, which is given by

$$\max(\Delta R(t)) = R(t) - R_{fuse_{max}} \tag{17}$$

5. Mazes under-test

In order to prove the behavior of the memristive grid in maze solving, this section presents several cases that have been ordered as follows:

- Mazes with a single solution
- Mazes with multiple solutions

- Maze with two inputs and two outputs
- An octogonal maze with three inputs and a single output
- A 3D maze

5.1 Single-solution mazes

The first set to be solved consists of three mazes with a single-entrance and single-output and the solution is given by a unique path.

5.1.1 The 5×5 *maze*

The first maze, from **Figure 12a**, is treated in full with the aim of highlighting the details of the solution procedure. The first stage of the procedure yields the memristive grid associated to the mapping of the maze, which is shown in **Figure 12b**. The resulting netlist of the memristive grid is then simulated in a transient analysis with HSPICE.

It can be noticed that there are 24 memristive fuses in the open paths of the maze. The electrical simulation is applied in order to measure the instantaneous resistance of the fuses. On the one hand, **Figure 13a** shows the transient behavior of the resistance of those fuses for the first $\frac{1}{5}$s. It can be noticed that all fuses start with the same resistance at $t = 0$, namely $R_{fuse_{init}}$. As a result, at $t = 0$, $\Delta R = 0$ for all fuses and the maze is not walked yet and the output display shows the open paths in white color, as shown in **Figure 13b**.

As time lapses, at $t = 0.197$s, only the fuses belonging to the solution path exhibit significant changes in their resistance. Here, the blue lines correspond to fuses outside the solution path, while the red lines correspond to fuses that belong to solution path. These changes are represented in the output display of **Figure 13c** for the same time in yellow. The solution path can already be distinguished.

On the other hand, **Figure 14a** shows $R(t)$ of the memristive fuses for $0 < t < 20$s. The red lines show that the fuses belonging to the solution path reached a maximum, while the blue lines remain in low levels of resistance, i.e., they belong to paths that finish in dead-ends.

Within this time-window, two snapshots of the output display have been taken at $t = 1.3929$s and $t = 3.7886$s—as depicted in the plots of **Figure 14b** and **c**, respectively. In the first display, the solution path is already highlighted in red with

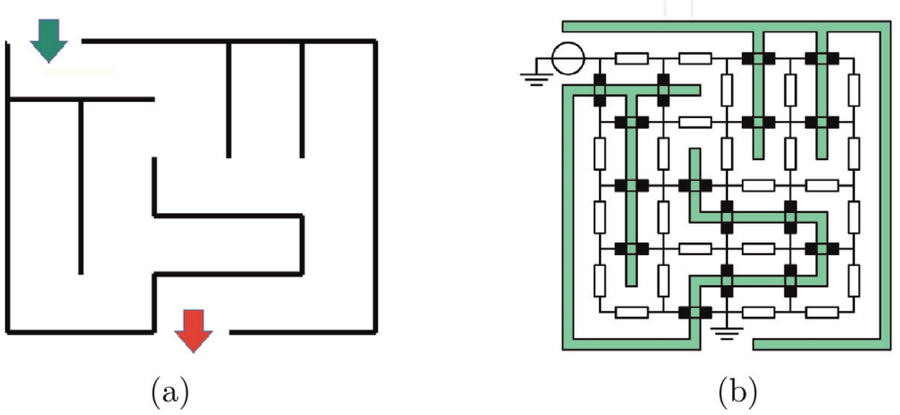

Figure 12.
Mapping the 5×5 *single-solution maze onto the memristive grid. (a) A* 5×5 *maze and (b) associated memristive grid.*

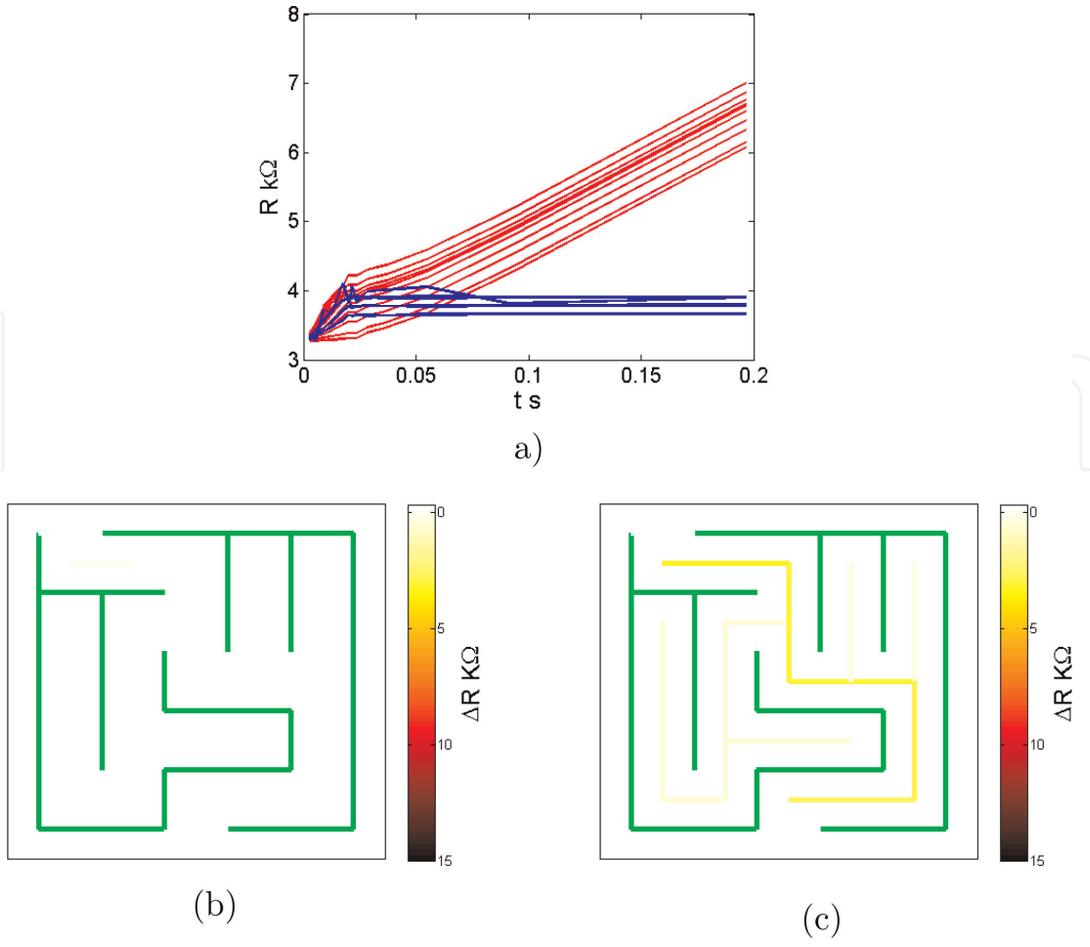

Figure 13.
Transient analysis of the maze in ***Figure 12*** *for small values of t. (a) R(t) of the fuses for* $0 < t < 0.197s$*, (b)* $t = 0$ s*, and (c)* $t = 0.197$ s.

fuses having a value of $\Delta R \approx 8.0k\Omega$. At $t = 3.7886$s, the fuses of the solution path show $\Delta R = 15.0k\Omega$.

In summary, it can be concluded that the memristive grid achieves the solution of the maze in a parallel processing by calculating the resistance of the fuses simultaneously. The progress of the solution procedure can be regarded as tracking the dynamic behavior of ΔR, which directly points out the solution path of the maze. On top of this, the output display allows us to visualize this procedure with the help of a color scale.

5.1.2 The 10×10 *and* 15×15 *mazes*

The memristive grid has also been applied to single-solution mazes that have larger sizes. The first maze is of 10×10 dimension and it is depicted in **Figure 15a** showing these mazes.

The second case is a 15×15 maze, which is shown altogether with its solution in **Figure 16**.

5.2 Multiple solutions mazes

The second set to be solved consists of three mazes that have solutions with multiple paths.

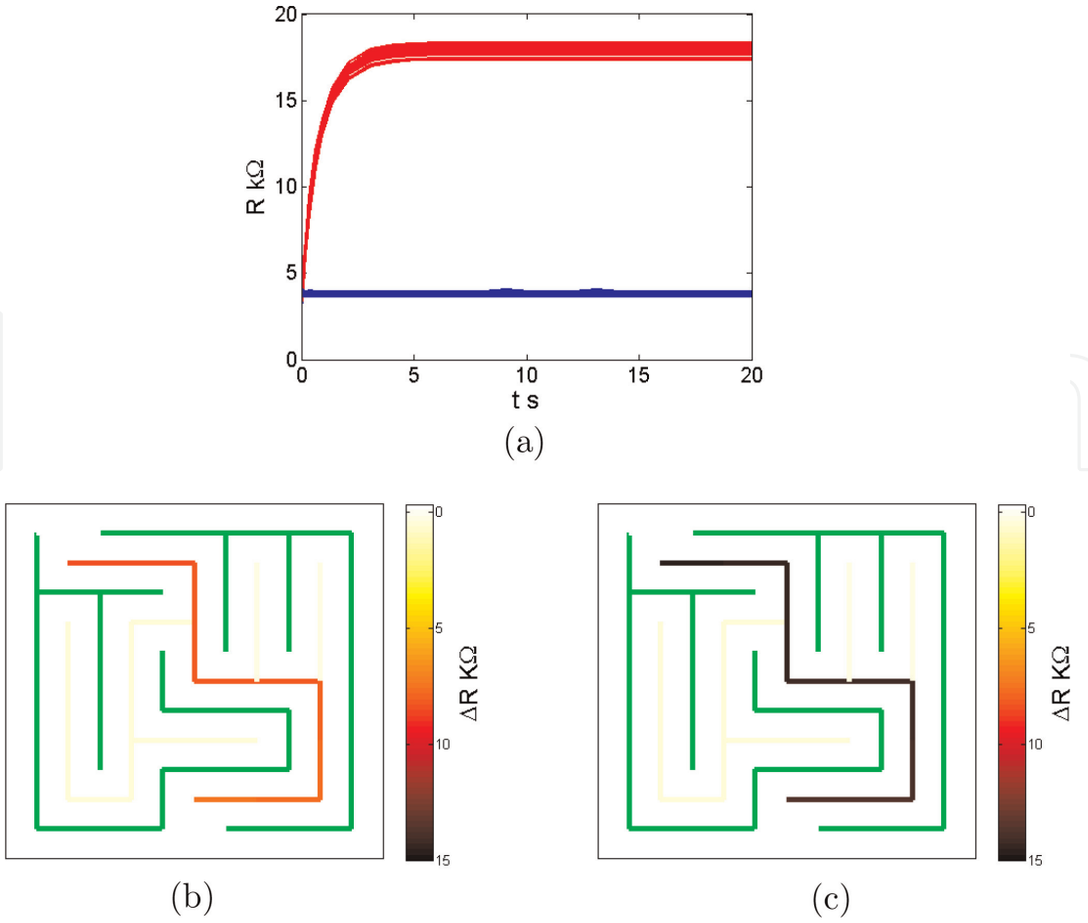

Figure 14.
Transient analysis of the maze in ***Figure 12*** *for larger values of t. (a) $R(t)$ of the fuses for $0 < t < 20$s, (b) $t = 1.3929$ s, and (c) $t = 3.7886$ s.*

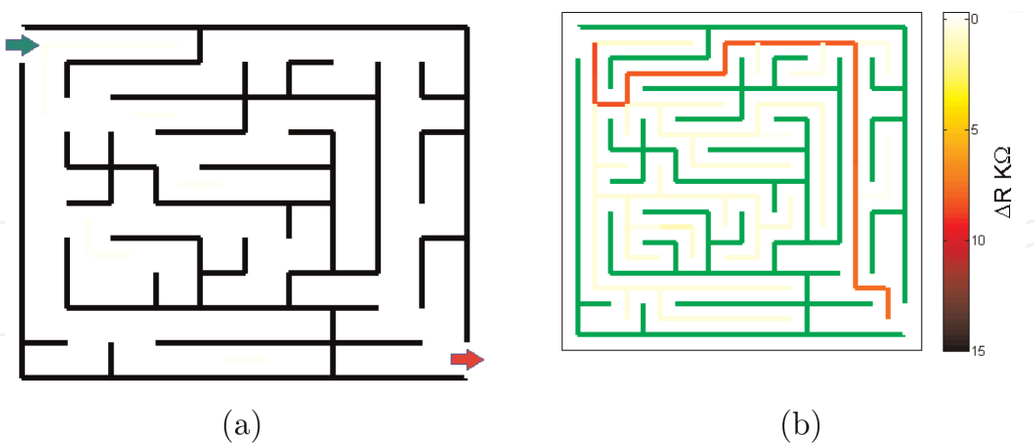

Figure 15.
10×10 *maze and solution at* $t = 1.3929$*s.*

5.2.1 The 5×5 *maze with multiple solutions*

This maze is shown in **Figure 17**. It is a very simple example that is explained to some extent in order to illustrate the procedure for finding the paths that constitute the solutions.

After carrying out the transient simulation, the resistance of the memristive fuses is obtained. **Figure 18a** shows $R(t)$ for $0 < t < 0.65$s. Herein, the attention is focused only on the resistance of the fuses belonging to the solution paths.

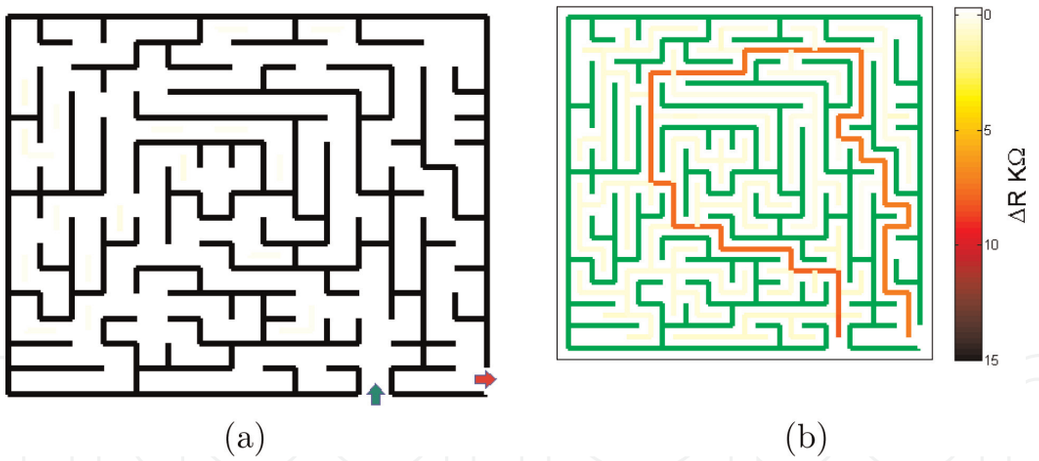

Figure 16.
15 × 15 maze and solution at t = 3.7886s.

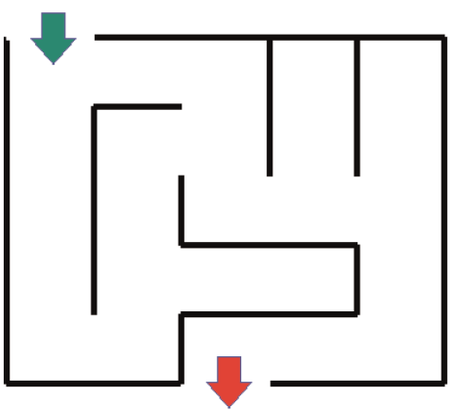

Figure 17.
A 5 × 5 double-solution maze.

Furthermore, the red lines show a steepest behavior which is a result that the red lines are associated to the fuses belonging to the solution with the shortest path. Besides, the blue lines are associated to fuses for the second solution path.

It can be observed that all paths start from $R_{fuse_{init}}$ when $t = 0$, i.e., the maze has not yet been walked—as given in the display of **Figure 18b**. After 0.2204 s, both solutions paths are already distinguishable, but the shortest path exhibits higher ΔR, which denoted by the darkest yellow tones in **Figure 18c**. After a while, at $t = 0.638$, the solution given by the shortest path is perfectly differentiable from the other solution, which can be compared by using the color bar.

After a larger sweep of time, the resistances of the fuses for both solutions have coalesced into an asymptotic level, which is the maximum value of the resistance at $t = 20$s—as shown in **Figure 19**.

5.2.2 Other mazes with multiple solutions

In this paragraph, two case studies are presented. The first one is the maze shown in **Figure 20a**, which is a 10 × 10 maze that has a single entrance and a single exit, but there are four possible solution paths.

A snapshot at 1.901 s has been taken—see **Figure 20b**. The four solution paths are visible in different colors. The shortest path is shown in red exhibiting the highest ΔR at the time of evaluation. On the opposite, the solution with the longest path is given in pale yellow. This example shows the usefulness of the color palette

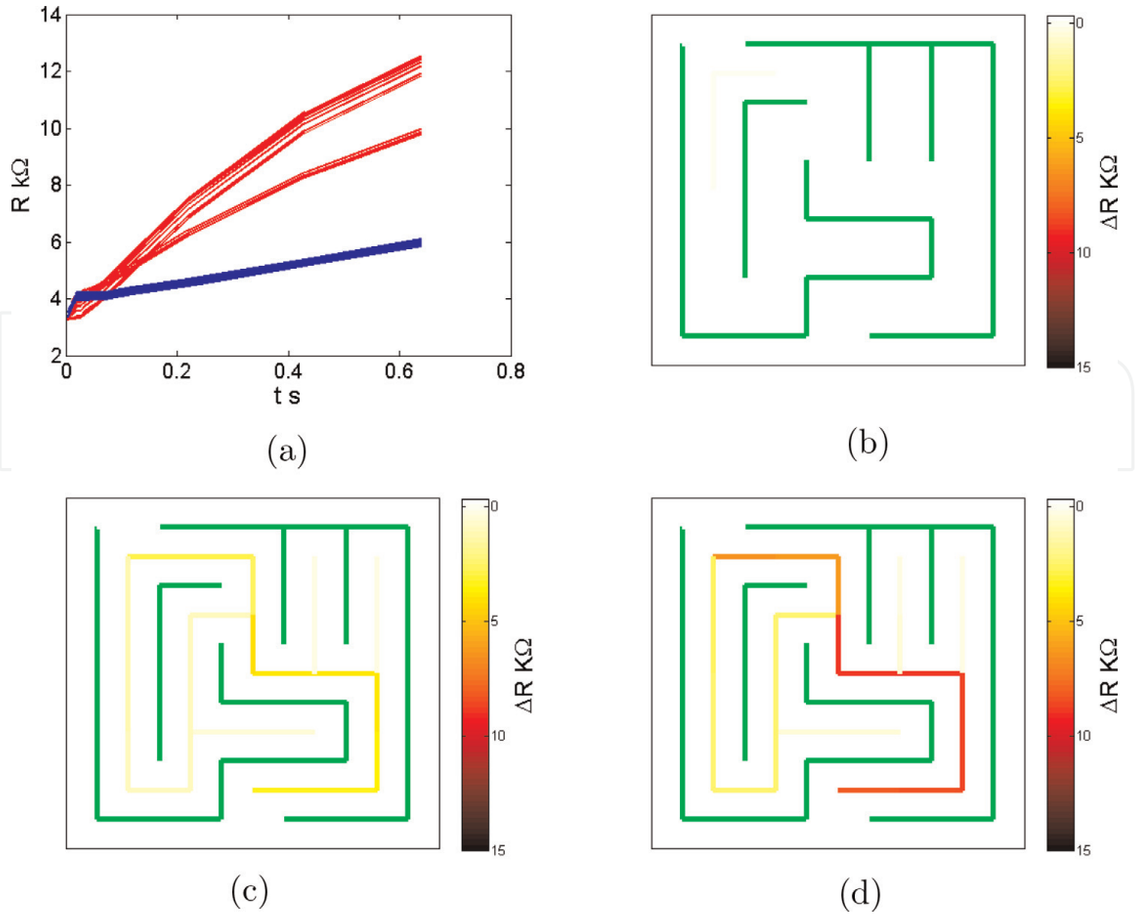

Figure 18.
Progress of the solution search for small t for the maze in ***Figure 17****. (a) $R(t)$ for $0 < t < 0.65s$, (b) $t = 0s$, (c) $t = 0.2204s$, and (d) $t = 0.638$ s.*

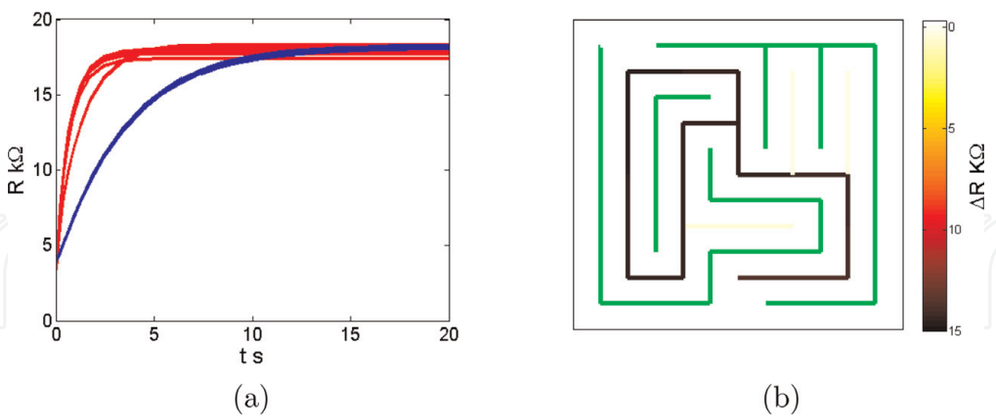

Figure 19.
Transient analysis for larger values of t. (a) $R(t)$ and (b) display at $t = 20s$.

of the output display on its full extent, because all possible solution paths are visible and it gives more insight on the progress of the solution procedure. After a long time, all the solutions reach the same resistance value as shown in **Figure 20c**.

The second example of this paragraph is a maze with two entrances and two exits that is shown in **Figure 21a**. We show in **Figure 21b** a snapshot taken at 1.0276 s. At this point, the memristive grid has been able to find both shortest paths for the solutions between the entrances and the outputs. After a while, at $t = 8.0716s$, the output display shows the connection between both paths—as given in **Figure 21c**.

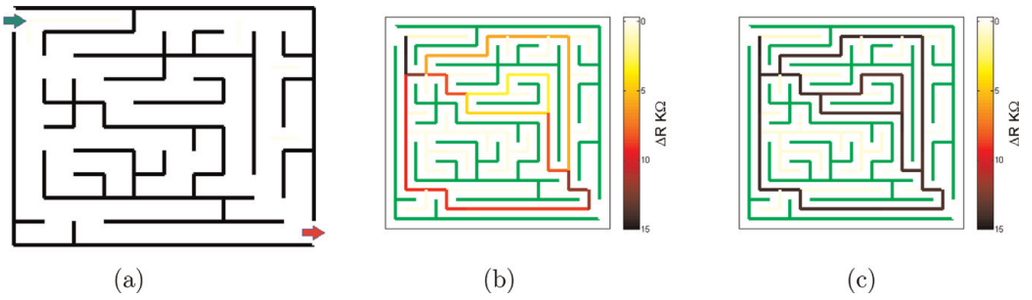

Figure 20.
Multiple-solution maze with one entrance and one exit. (a) Maze, (b) $t = 1.901s$, and (c) $t = 20s$.

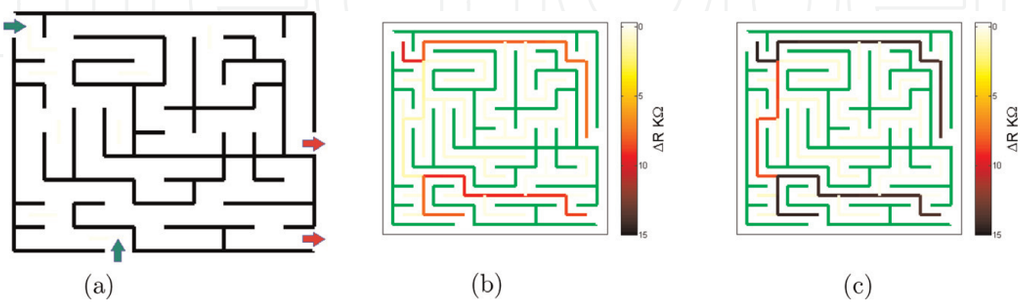

Figure 21.
Multiple-solution maze with two entrances and one exits. (a) Maze, (b) $t = 1.0276s$, and (c) $t = 8.0716s$.

5.3 Octogonal maze

A nonrectangular maze is given in **Figure 22a**, which is an octogonal concentric maze with three entrances and with the output goal in the center of the maze. The three entrances are denoted as **S**, **E**, and **NW**. Entrances **E** and **NW** cannot reach the solution, while entrance **S** does. Given the impossibility of the output display for dealing with nonrectangular mazes, the octogonal maze is converted into an isomorphic view that is given in **Figure 22b** that shows the solution path in red.

5.4 A 3D maze

In order to illustrate that the memristive grid is able to deal with a three-dimensional maze, a three-layer maze is solved. For sake of readiness, **Figure 23** shows the maze in separated levels in a puzzle-fashion. The ball on the top-layer

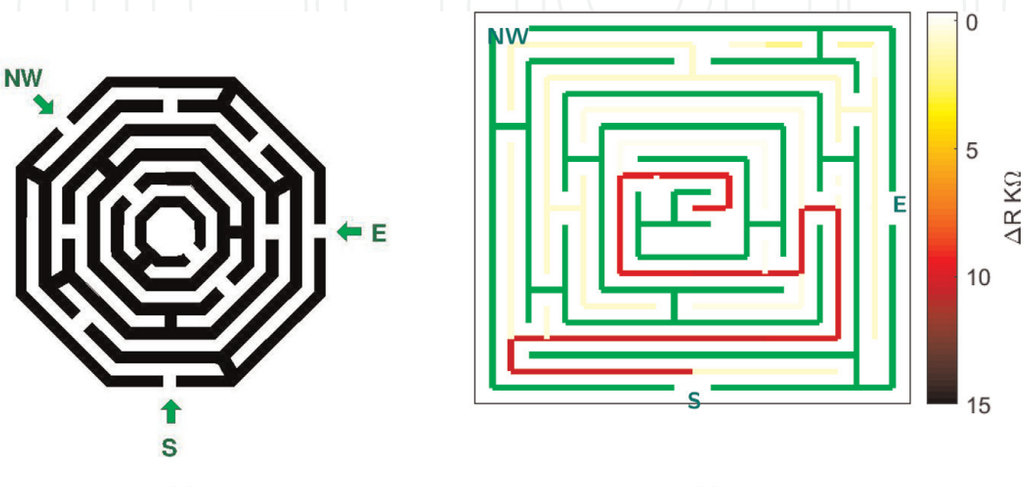

Figure 22.
Octogonal maze and solution.

Figure 23.
A 3D-maze.

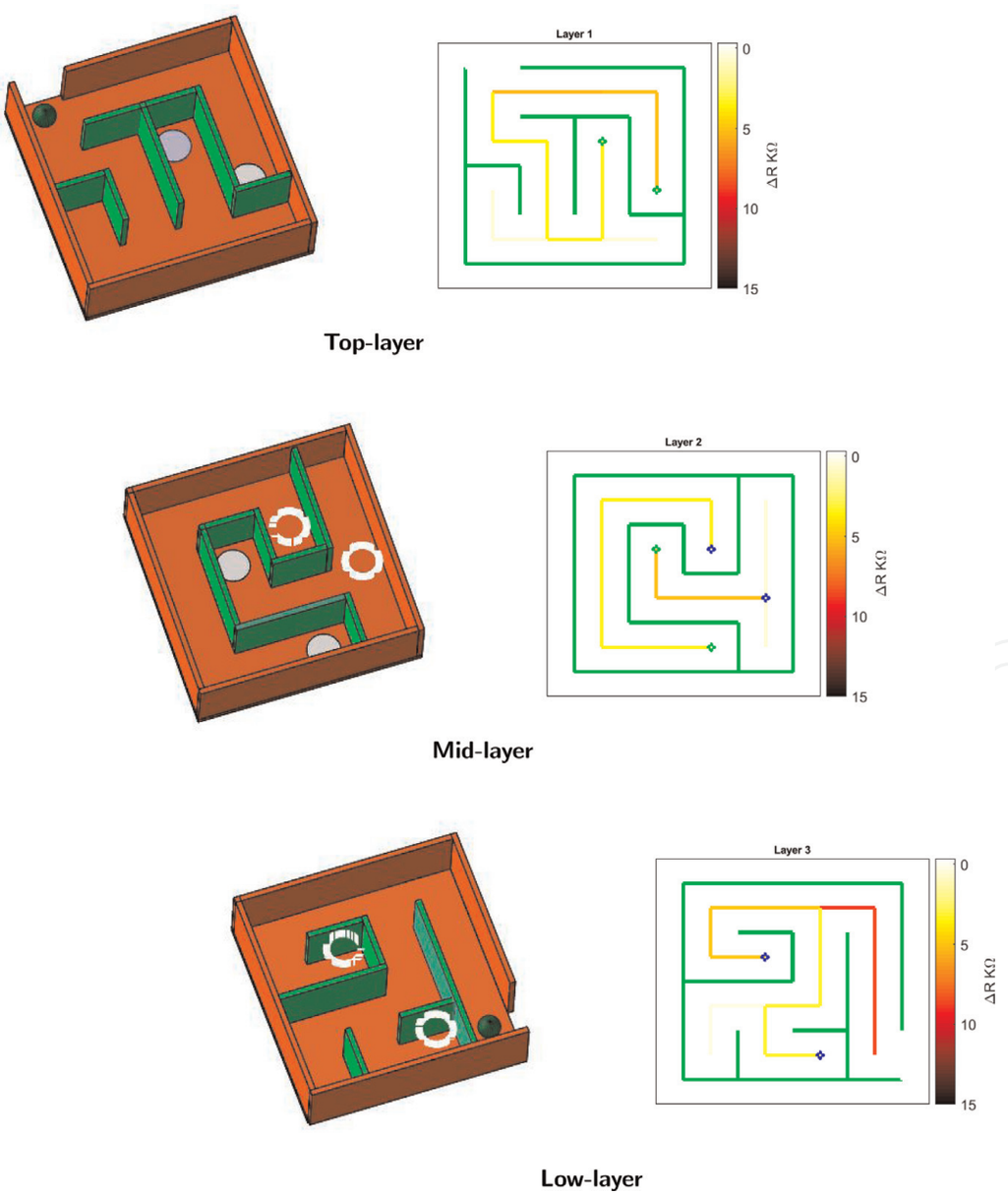

Figure 24.
Solutions of the 3D-maze.

indicates the starting point of the maze, while the ball in the low-layer points out to the output of the maze. The layers communicate each other with holes that are depicted as circles on the floor and disks on the roof of them.

The memristive grid that describes this maze counts 16 nodes per layer which yields a total of 48 nodes. Every layer possesses 24 branches (the external walls do not count) plus four inter-layer branches, i.e., 78 memristive fuses to describe the maze.

Finally, the output display given in **Figure 24** shows the progress of the solution procedure.

6. Code for the model

In the following, the code for the memristor model as used within HSPICE is given.

```
*Charge-controlled models
*INAOE, summer 2018
*Yojanes Rodríguez
.LIB MemModels
*-------------------------------------------------------
*HPMQ Joglekar k=5 01
.SUBCKT HPMQK501N N+ N-
.PARAM Xo=0.99
.PARAM mu=10f
.PARAM eta=-1
.PARAM Roff=16e3
.PARAM Ron=100
.PARAM Delta=10n
.PARAM kappa='Ron*mu/(POW(Delta,2))'
.PARAM Pol1='-(256/45)*POW(Xo,10)+32*POW(Xo,9)-(576/7)*POW(Xo,8)+128*POW
(Xo,7)-
+(672/5)*POW(Xo,6)+(504/5)*POW(Xo,5)-56*POW(Xo,4)+24*POW(Xo,3)-9*POW
(Xo,2)-Xo'
.PARAM Pol2='(256/45)*POW(Xo,10)-(224/9)*POW(Xo,9)+(352/7)*POW(Xo,8)-(1280/
21)*POW(Xo,7)+
+(736/15)*POW(Xo,6)-(136/5)*POW(Xo,5)+(32/3)*POW(Xo,4)-(8/3)*POW(Xo,3)+
+POW(Xo,2)-(1441/315)*Xo+1126/315'
.PARAM Pol3='-9*POW(Xo,2)+18*Xo-9'
.PARAM Pol4='-24*POW(Xo,3)+72*POW(Xo,2)-72*Xo+24'
.PARAM Pol5='-56*POW(Xo,4)+224*POW(Xo,3)-336*POW(Xo,2)+224*Xo-56'
.PARAM Pol6='-(504/5)*POW(Xo,5)+504*POW(Xo,4)-1008*POW(Xo,3)+1008*POW
(Xo,2)-504*Xo+504/5'
.PARAM Pol7='-(672/5)*POW(Xo,6)+(4032/5)*POW(Xo,5)-2016*POW(Xo,4)
+2688*POW(Xo,3)-
+2016*POW(Xo,2)+(4032/5)*Xo-(672/5)'
.PARAM Pol8='-128*POW(Xo,7)+896*POW(Xo,6)-2688*POW(Xo,5)+4480*POW(Xo,4)-
+4480*POW(Xo,3)+2688*POW(Xo,2)-896*Xo+128'
.PARAM Pol9='-(576/7)*POW(Xo,8)+(4608/7)*POW(Xo,7)-2304*POW(Xo,6)
+4608*POW(Xo,5)-
+5760*POW(Xo,4)+4608*POW(Xo,3)-2304*POW(Xo,2)+(4608/7)*Xo-576/7'
.PARAM Pol10='-32*POW(Xo,9)+288*POW(Xo,8)-1152*POW(Xo,7)+2688*POW(Xo,6)-
4032*POW(Xo,5)+
+4032*POW(Xo,4)-2688*POW(Xo,3)+1152*POW(Xo,2)-288*Xo+32'
.PARAM Pol11='-(256/45)*POW(Xo,10)+(512/9)*POW(Xo,9)-256*POW(Xo,8)+(2048/
```

```
3)*POW(Xo,7)-
+(3584/3)*POW(Xo,6)+(7168/5)*POW(Xo,5)-(3584/3)*POW(Xo,4)+
+(2048/3)*POW(Xo,3)-256*POW(Xo,2)+(512/9)*Xo-256/45'

*Integrator
Ecur Ni 0 VOL = 'I(Rmem)'
R Ni Na 1k
C Na No 1m
Eop No GND GND Na 1Meg

Echarge charge GND No GND -1

Rmem N+ N- R='(V(charge)>0)?(+(256/45)*POW(Xo,10)*exp(-200*kappa*V
(charge))-
+32*POW(Xo,9)*exp(-180*kappa*V(charge))+(576/7)*POW(Xo,8)*exp(-160*kappa*V
(charge))-
+128*POW(Xo,7)*exp(-140*kappa*V(charge))+(672/5)*POW(Xo,6)*exp(-
120*kappa*V(charge))-
+(504/5)*POW(Xo,5)*exp(-100*kappa*V(charge))+56*POW(Xo,4)*exp(-80*kappa*V
(charge))-
+24*POW(Xo,3)*exp(-60*kappa*V(charge))+9*POW(Xo,2)*exp(-40*kappa*V
(charge))+
+(Pol1)*exp(-20*kappa*V(charge)))*(Roff-Ron)+Roff:((Pol2)*exp(20*kappa*V
(charge))+
+(Pol3)*exp(40*kappa*V(charge))+(Pol4)*exp(60*kappa*V(charge))+
+(Pol5)*exp(80*kappa*V(charge))+(Pol6)*exp(100*kappa*V(charge))+
+(Pol7)*exp(120*kappa*V(charge))+(Pol8)*exp(140*kappa*V(charge))+
+(Pol9)*exp(160*kappa*V(charge))+(Pol10)*exp(180*kappa*V(charge))+
+(Pol11)*exp(200*kappa*V(charge)))*(Roff-Ron)+Ron'
.ENDS
*-------------------------------------------------------
.ENDL MemModels
```

7. Conclusions

A specially tailored memristive grid has been used as an analog processor for solving mazes. The memristives branches of the grid (fuses) are formed by an anti-series connection of two memristors and a switch. On one side, we have introduced a family of symbolic models for the memristor that are defined by charge-controlled functions. The fact that the models are charge-controlled allows us to monitor the velocity of the variation of the equivalent memristance of the fuses by carrying out a transient analysis with HSPICE. It is worth to mention that the model has been recast in VERILOG-A. On the other side, with the aim of producing a more realistic scenario, the switches are implemented by a transmission gate in CMOS technology. In this form, the resulting grid is in fact a hybrid CMOS-Memristor circuit.

The simulation flow-work is formed by an input stage developed in MATLAB, the electric simulation in HSPICE and the output stage again in MATLAB. The input stage is responsible for mapping the structure of the maze onto the memristive grid. The outcome of this stage is an input file with the netlist of the grid. The intermediate stage executes the transient simulation. The output stage is used to display the variation of the resistance of the fuses and it literally draws the solution path of the

maze. The solution is found by sensing the variations of the resistance of the fuses that belong to the path, which implies that the memristive grid achieves the shortest path algorithm.

Finally, the maze grid has proven its reliability in solving mazes with different levels of complexity. A series of examples has been analyzed: single-solution mazes, multiple-solution mazes, and a 3D maze.

Author details

Arturo Sarmiento-Reyes* and Yojanes Rodríguez Velásquez
Electronics Department, National Institute for Astrophysics, Optics and Electronics, San Andrés Cholula, Puebla, Mexico

*Address all correspondence to: jarocho@inaoep.mx

References

[1] Kern H, Saward J, Schons M, Clay AH, Thomson SB, Velder KA. Through the Labyrinth: Designs and Meanings Over 5,000 Years. Art and Design Series. New York: Prestel Publishing; 2000

[2] Barnes CA. Memory deficits associated with senescence: A neurophysiological and behavioral study in the rat. Journal of Comparative and Physiological Psychology. 1979;**93**(1):74

[3] Olton DS, Samuelson RJ. Remembrance of places passed: Spatial memory in rats. Journal of Experimental Psychology: Animal Behavior Processes. 1976;**2**(2):97

[4] Morris R. Developments of a water-maze procedure for studying spatial learning in the rat. Journal of Neuroscience Methods. 1984;**11**(1): 47-60

[5] Dracopoulos DC. Robot path planning for maze navigation. In: The 1998 IEEE International Joint Conference on Neural Networks Proceedings, IEEE World Congress on Computational Intelligence; Vol. 3. IEEE; 1998. pp. 2081-2085

[6] Lumelsky VJ. A comparative study on the path length performance of maze-searching and robot motion planning algorithms. IEEE Transactions on Robotics and Automation. 1991;**7**(1): 57-66

[7] Werbos PJ, Pang X. Generalized maze navigation: SRN critics solve what feedforward or Hebbian nets cannot. In: IEEE International Conference on Systems, Man, and Cybernetics; Vol. 3. IEEE; 1996. pp. 1764-1769

[8] Milková E, Slaby A. Graph algorithms in mutual contexts. In: 7th WSEAS International Conference Proceedings on Mathematics and Computers in Science and Engineering; Vol. 1. World Scientific and Engineering Academy and Society; 2008. pp. 721-726

[9] Bondy JA, Murty USR. Graph Theory with Applications. Vol. 290. London: Macmillan; 1976

[10] Abelson H, DiSessa AA. Turtle Geometry: The Computer as a Medium for Exploring Mathematics. Cambridge, Massachusetts: MIT Press; 1986

[11] Müller H. A one-symbol printing automaton escaping from every labyrinth. Computing. 1977;**19**(2): 95-110

[12] Vourkas I, Stathis D, Sirakoulis G. Massively parallel analog computing: Ariadnes thread was made of memristors. IEEE Transactions on Emerging Topics in Computing. 2015; **6**(1):145-155

[13] Ye Z,Wu SHM, Prodromakis T. Computing shortest paths in 2D and 3D memristive networks. In: Adamatzky A, Chua LO. Memristor Networks. Basel, Switzerland: Springer; 2014. pp. 537-552

[14] Pershin YV, Di Ventra M. Solving mazes with memristors: A massively parallel approach. Physical Review E. 2011;**84**(4):046703

[15] Chua L. Memristor—The missing circuit element. IEEE Transactions on Circuit Theory. 1971;**18**(5):507-519

[16] Chua LO, Kang S-M. Memristive devices and systems. Proceedings of the IEEE. 1976;**64**(2):209-223

[17] Strukov DB, Snider GS, Stewart DR, Williams RS. The missing memristor found. Nature. 2008;**453**(7191):80-83

[18] Indiveri G, Linares-Barranco B, Legenstein R, Deligeorgis G,

Prodromakis T. Integration of nanoscale memristor synapses in neuromorphic computing architectures. Nanotechnology. 2013;**24**(38):384010

[19] Vittoz EA. Future of analog in the VLSI environment. In: IEEE International Symposium on Circuits and Systems. IEEE; 1990. pp. 1372-1375

[20] Chua L. Memristor, Hodgkin-Huxley, and edge of chaos. In: Adamatzky A, Chua LO. Memristor Networks. Basel, Switzerland: Springer; 2014. pp. 67-94

[21] Jo SH, Chang T, Ebong I, Bhadviya BB, Mazumder P, Lu W. Nanoscale memristor device as synapse in neuromorphic systems. Nano Letters. 2010;**10**(4):1297-1301

[22] Naous R, Al-Shedivat M, Salama KN. Stochasticity modeling in memristors. IEEE Transactions on Nanotechnology. 2016;**15**(1):15-28

[23] Pershin YV, Di Ventra M. Memcomputing: A computing paradigm to store and process information on the same physical platform. In: 2014 International Workshop on Computational Electronics (IWCE). IEEE; 2014. pp. 1-2

[24] Sarmiento-Reyes A, Hernández-Martínez L, Vázquez-Leal H, Hernández-Mejía C, Diaz Arango GU. A fully symbolic homotopy-based memristor model for applications to circuit simulation. Analog Integrated Circuits and Signal Processing. 2015; **85**(1):65-80

[25] He J-H. Homotopy perturbation technique. Computer Methods in Applied Mechanics and Engineering. 1999;**178**(3):257-262

[26] Vazquez-Leal H. Generalized homotopy method for solving nonlinear differential equations. Computational and Applied Mathematics. 2014;**33**(1): 275-288

[27] He J-H. Comparison of homotopy perturbation method and homotopy analysis method. Applied Mathematics and Computation. 2004;**156**(2):527-539

[28] Joglekar YN, Wolf SJ. The elusive memristor: Properties of basic electrical circuits. European Journal of Physics. 2009;**30**(4):661

[29] Sarmiento-Reyes A, Velásquez YR. Chapter 5: Charge-controlled memristor grid for edge detection. In: Ciufudean C, editor. Advances in Memristor Neural Networks. London, United Kingdom: InTechOpen; 2018. pp. 91-113

[30] Velásquez YAR. Development of an analytical model for a charge-controlled memristor and its applications [Master's thesis]. Puebla, Mexico: National Institute for Astrophysics, Optics and Electronics (INAOE); 2017

[31] Recski A. Matroid Theory and Its Applications. Berlin, Germany: Spring-Verlag; 1989

[32] Hodges DA, Jackson HG. Analysis and Design of Digital Integrated Circuits. Boston, USA: McGraw-Hill; 1988