Nat Comput (2012) 11:381–386
DOI 10.1007/s11047-011-9287-4

Designing a new software tool for Digital Imagery based on P systems

Daniel Díaz-Pernil · Miguel A. Gutiérrez-Naranjo · Helena Molina-Abril · Pedro Real

Published online: 4 September 2011

Abstract In this paper we present a new software tool for dealing with the problem of segmentation in Digital Imagery. The implementation is inspired in the design of a tissue-like P system which solves the problem in constant time due the intrinsic parallelism of Membrane Computing devices.

Keywords Membrane computing · Digital Imagery · Segmentation

1 Introduction

Nature is a big inspiration source for designing solutions to a broad panoply of problems. Natural Computing studies computational paradigms inspired from various well known natural phenomena in physics, chemistry and biology[1]. It abstracts the way in which nature acts, conceiving new computing models. The field is growing rapidly and there are many open research lines based on nature. Among them, *Cellular Automata* (von Newmann 1966) conceived by Ulam and von Newman as a spatial distribution of cells able to reproduce the behavior of complex systems; *Genetic Algorithms* introduced by Holland (1992) which is inspired by natural evolution and selection in order to find a good solution in a large set of feasible candidate solutions; *Neural Networks* introduced by McCulloch and Pitts (1988) based on the interconnections of neurons in the brain; *DNA-based molecular computing*, that was born when Adleman (1994) published a solution to an instance of the Hamiltonian path problem by manipulating DNA strands in a lab; *Swarm Intelligence* (Engelbrecht 2005) based on the behavior and communication of mobile organisms as ants or bees acting in the environment; *Artificial Immune Systems* (de Castro and Timmis 2002) based on the natural immune system of biological organisms; *Amorphous Computing* (Abelson et al. 2000) inspired from the development of morphogenesis in biological organisms or *Membrane Computing* (Păun 2000, 2002) based on the functioning and morphology of living cells and tissues.

All these computational paradigms have in common the use of an alternative way of encoding the information, adapted to the bio-inspired substrate and the use of intrinsic parallelism of natural processes.

In this paper we present a bio-inspired software for solving the Segmentation Problem in Digital Imagery. Segmentation in computer vision (see Shapiro and Stockman 2001), refers to the process of partitioning a digital image into multiple segments (sets of pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to

D. Díaz-Pernil · H. Molina-Abril · P. Real
Research Group on Computational Topology and Applied Mathematics, University of Seville, Seville, Spain
e-mail: sbdani@us.es

H. Molina-Abril
e-mail: habril@us.es

P. Real
e-mail: real@us.es

M. A. Gutiérrez-Naranjo (✉)
Research Group on Natural Computing,
Department of Computer Science and Artificial Intelligence,
University of Seville, Seville, Spain
e-mail: magutier@us.es

[1] An introduction on Natural Computing can be found in Kari and Rozenberg (2008).

locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain visual characteristics.

Segmentation in Digital Imagery has several features which make it suitable for techniques inspired by nature. One of them is that it can be parallelized and locally solved. Regardless how large is the picture, the segmentation process can be performed in parallel in different local areas of it. Another interesting feature is that the basic necessary information can be easily encoded by bio-inspired representations.

In the literature, one can find several attempts for bridging problems from Digital Imagery with Natural Computing as the works by Subramanian and coworkers (2003a, b] or the work by Chao and Nakayama (1996) where Natural Computing and Algebraic Topology are linked by using Neural Networks (extended Kohonen mapping). In this paper, we will use an information encoding and techniques borrowed from Membrane Computing.

Membrane Computing is a theoretical model of computation inspired by the structure and functioning of cells as living organisms able to process and generate information. The computational devices are called *P systems* (Păun 2000). Roughly speaking, a P system consists of a membrane structure, in the compartments of which one places multisets of objects which evolve according to given rules. In the most extended model, the rules are applied in a synchronous non-deterministic maximally parallel manner, but some other semantics are being explored[2].

According to their architecture, these models can be split into two sets: P systems such that their membrane structure is a tree-like graph, called *cell-like P systems* and P systems whose membrane structure is a general graph. In this second group we can find *tissue-like P systems* and *spiking neural P systems*. This paper is devoted to the second approach: tissue-like P systems. In Christinal et al. (2009a, b, 2010) started a new bio-inspired research line where the power and efficiency of tissue-like P systems (Díaz-Pernil et al. 2008, 2009) were applied to topological processes for 2D and 3D digital images. In this paper, we present a new software tool to segment 2D digital images based in the works of Christinal et al. just mentioned. This tool simulates the behavior of the tissue-like P systems described in Christinal et al. (2009a) and allows us to work with images in JPG format.

Simulation of different variants of P systems have been widely studied in the last years. Since there do not exist implementations of P systems in vivo nor in vitro, the natural way to explore the behavior of designed P systems is to simulate it in conventional computers. A short description of some of these simulators can be found in Díaz-Pernil et al. (2010), Gutiérrez-Naranjo et al. (2006). In (Borrego-Ropero et al. 2007), a first simulator for tissue-like P systems was presented. Currently, a big effort is being developed in the *P-lingua project* (Díaz-Pernil et al. 2008), by combining an efficient simulation engine with an ad hoc description language.

The paper is organized as follows: firstly, we present our bio-inspired formal framework. Next, we present the family of tissue-like P systems used to obtain a segmentation of a 2D digital image. In Sect. 4 we introduce our software tool and illustrate its use with some examples. Finally, some conclusions are presented.

2 Formal framework: tissue-like P systems

Tissue-like P systems were presented by Martín-Vide et al. (2002). They have two biological inspirations (see Martín-Vide 2003): intercellular communication and cooperation between neurons. The common mathematical model of these two mechanisms is a network of processors dealing with symbols and communicating these symbols along channels specified in advance.

The main features of this model, from the computational point of view, are that the membrane structure is a general graph and the objects in the environment are available in an arbitrarily large amount of copies.

Formally, a *tissue-like P system* with input of degree $q \geq 1$ is a tuple

$$\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, i_\Pi, o_\Pi),$$

where

(1) Γ is a finite *alphabet*, whose symbols will be called *objects*,
(2) $\Sigma (\subset \Gamma)$ is the input alphabet,
(3) $\mathcal{E} \subseteq \Gamma$ (the objects in the environment),
(4) $w_1, \ldots, w_q$ are strings over Γ representing the multisets of objects associated with the cells at the initial configuration,
(5) $\mathcal{R}$ is a finite set of communication rules of the following form:

$$(i, u/v, j)$$

for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j$, $u, v \in \Gamma^*$,
(6) $i_\Pi \in \{0, 1, 2, \ldots, q\}$ is the input cell,
(7) $o_\Pi \in \{0, 1, 2, \ldots, q\}$ is the output cells.

A tissue-like P system of degree $q \geq 1$ can be seen as a set of q cells (each one consisting of an elementary membrane) labelled by $1, 2, \ldots, q$. We will use 0 to refer to the label of the environment, i_Π denotes the input cell and

[2] We refer to Păun (2002) for basic information in this area, to Păun (2010) for a comprehensive presentation and the web site P system web page for the up-to-date information.

o_Π denotes the output cell (which can be the region inside a cell or the environment).

The strings $w_1, \ldots, w_q$ describe the multisets of objects placed in the q cells of the system. We interpret that $\mathcal{E} \subseteq \Gamma$ is the set of objects placed in the environment, each one of them available in an arbitrarily large amount of copies.

The communication rule $(i, u/v, j)$ can be applied over two cells labelled by i and j such that u is contained in cell i and v is contained in cell j. The application of this rule means that the objects of the multisets represented by u and v are interchanged between the two cells. Note that if either $i = 0$ or $j = 0$ then the objects are interchanged between a cell and the environment.

Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way (a universal clock is considered). In one step, each object in a membrane can only be used for one rule (non-deterministically chosen when there are several possibilities), but any object which can participate in a rule of any form must do it, i.e., in each step we apply a maximal set of rules.

A *configuration* is an instantaneous description of the system Π. Given a configuration, we can perform a computation step and obtain a new configuration by applying the rules in a parallel manner as it is shown above. A *computation* is a sequence of computation steps such that either it is infinite or it is finite and the last step yields a halting configuration (i.e., no rules can be applied to it). Then, a computation halts when the system reaches a halting configuration.

3 Segmenting digital images in constant time

In this section, we segment images based on edge-based segmentation. It consists on finding boundaries of regions which are sufficiently different from each other. There exist different techniques to segment an image. Some of them are clustering (Wang et al. 2005), histogram-based methods (Tobias and Seara 2002), watershed transformation methods (Yazid and Arof 2008), graph partitioning (Yuan et al. 2009) and image pyramids methods (Kropatsch et al. 2007). Some of the practical applications of image segmentation are medical imaging (Wang et al. 2005), objects classification and face recognition (Kim et al. 1998). We define a family of tissue-like P systems to segment 2D images.

3.1 A family of tissue-like P systems for a 2D segmentation

We can divide the image in multiple pixels forming a network of points of $\mathbb{N}^2$. Let $\mathcal{C} \subseteq \mathbb{N}$ be the ordered set of all colors in the given 2D image. Moreover, we will suppose each pixel is associated with a color of the image. Then we can codify the pixel (i, j) with associated color $a \in \mathcal{C}$ by the object a_{ij}.

The following question is to decide which pixel is adjacent to a given one. We have decided to use in this paper the 4-adjacency (Rosenfeld 1970, 1979). In this case, each pixel has four (horizontal and vertical) neighbors. The extension to different adjacencies is straightforward.

At this point, we want to find the border cells of the different color regions that are within the image. Then, for each image with $n \times m$ pixels $(n, m \in \mathbb{N})$ we will construct a tissue-like P system whose input is given by the objects a_{ij} codifying a pixel, with $a \in \mathcal{C}$. The output of the system is given by the objects that appear in the output cell when the system stops.

Based on that, we define a family of tissue-like P systems to perform an edge-based segmentation to a 2D image.

For each $n, m \in \mathbb{N}$ we consider the tissue-like P system

$$\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, w_2, \mathcal{R}, i_\Pi, o_\Pi)$$

defined as follows:

(a) $\Gamma = \Sigma \cup \{\bar{a}_{ij} : 1 \leq i \leq n,\ 1 \leq j \leq m,\ a \in \mathcal{C}\}$
$\cup \{A_{ij} : 1 \leq i \leq n, 1 \leq j \leq m,\ A \in \mathcal{C}\}$,
(b) $\Sigma = \{a_{ij} : a \in \mathcal{C},\ 1 \leq i \leq n,\ 1 \leq j \leq m\}$,
(c) $\mathcal{E} = \Gamma - \Sigma$,
(d) $w_1 = w_2 = \emptyset$,
(e) R is the following set of communication rules:

(1) $(1, a_{ij}b_{kl}/\bar{a}_{ij}A_{ij}b_{kl}, 0)$, for $a, b \in \mathcal{C}$, $a < b$, $1 \leq i, k \leq n$ and $1 \leq j, l \leq m$.

These rules are used when the image has two adjacent pixels with different associated colors (border pixels). Then, the pixel with lower associated color is marked and the system brings from the environment an object representing this marked pixel (edge pixel).

(2) $(1, \bar{a}_{ij}a_{ij+1}\bar{a}_{i+1j+1}b_{i+1j}/\bar{a}_{ij}\bar{a}_{ij+1}A_{ij+1}\bar{a}_{i+1j+1}b_{i+1j}, 0)$ for $a, b \in \mathcal{C}$, $a < b$, $1 \leq i \leq n-1$, $1 \leq j \leq m-1$.

$(1, \bar{a}_{ij}a_{i-1j}\bar{a}_{i-1j+1}b_{ij+1}/\bar{a}_{ij}\bar{a}_{i-1j}A_{i-1j}\bar{a}_{i-1j+1}b_{ij+1}, 0)$ for $a, b \in \mathcal{C}$, $a < b$, $2 \leq i \leq n$, $1 \leq j \leq m-1$.

$(1, \bar{a}_{ij}a_{ij+1}\bar{a}_{i-1j+1}b_{i-1j}/\bar{a}_{ij}\bar{a}_{ij+1}A_{ij+1}\bar{a}_{i-1j+1}b_{i-1j}, 0)$ for $a, b \in \mathcal{C}$, $a < b$, $2 \leq i \leq n$, $1 \leq j \leq m-1$.

$(1, \bar{a}_{ij}a_{i+1j}\bar{a}_{i+1j+1}b_{ij+1}/\bar{a}_{ij}\bar{a}_{i+1j}A_{i+1j}\bar{a}_{i+1j+1}b_{ij+1}, 0)$ for $a, b \in \mathcal{C}$, $a < b$, $1 \leq i \leq n-1$, $1 \leq j \leq m-1$.

These rules mark with a bar the pixels which are adjacent to two pixels of the same color which were marked before, but with the condition that the marked objects are adjacent to another pixel with a different color. Moreover, an edge object representing the last marked pixel is brought from the environment.

(3) $(1, A_{ij}/\lambda, 2)$, for $1 \leq i \leq n$, $1 \leq j \leq m$. This rule is used to send the edge pixels to the output cell.

(f) $i_\Pi = 1$

(g) $o_\Pi = 2$.

3.2 An overview of the computation

Rules of type 1, in a parallel manner, identify the border pixels and bring the edge pixels from the environment. These rules need 4 steps to mark all the border pixels. From the second step, the rules of type 2 can be used with the first rules at the same time. So, in 4 more steps we can bring from the environment the edge pixels adjacent to two border pixels (as explained above). The P system can apply the first two types of rules simultaneously in some configurations, but it always applies the same number of these two types of rules because this number is given by the edge pixels (we consider 4-adjacency). Finally, the third type of rules are applied in the following step on the edge pixels appearing in the cell. So, with one more step we will have all the edge pixels in the output cells. Thus, we need only 9 steps to obtain an edge-based segmentation for an $n \times m$ image. Therefore, we can conclude that the problem of edge-segmentation in 2D images is solved in constant time with respect to the number of steps of any computation.

4 A software tool

In (Christinal et al. 2009), preliminary segmentation results were obtained using the *tissue simulator* developed in Borrego-Ropero et al. (2007). Such a *tissue simulator* follows one of the common features of the first generation of simulators for cell-like P systems, that is the lack of efficiency in favor of expressiveness. Therefore, experiments performed using this tool were extremely slow, and could only use synthetic images of at most 30×30 pixels.

In order to perform experiments with bigger images, a new software tool has been developed. This software makes possible the obtaining of segmented real images that have been partitioned following a membrane computing approach.

For optimal effectiveness and flexibility, the object oriented C++ programming language has been used in the implementation.

The software input consists of a digital 2D image. The image format can be any of the most common raster image formats (jpg, png, gif,…). Such image is provided to the P system as a set of objects a_{ij} where (i, j) covers the $n \times m$ array of pixels and a belongs to $\mathcal{C}$, the set of colors.

At the beginning, the input cell contains objects a_{ij} codifying the colored pixels from an 2D image (where a is the color value of the pixel, and i, j its coordinates).

As an output, the software provides a black image (with the same format as the input image), where the detected border pixels are white. In other words, pixels belonging to the output cell of the system will be considered white, and printed out in the output image.

Some experiments and results are shown in Figs. 1, 2 and 3. Figure 1 shows a geometrical 340×340 picture together with the output image of the software. Due to the sharp boundaries within the image, a precise segmentation result is obtained. In Fig. 2 a more complex image is

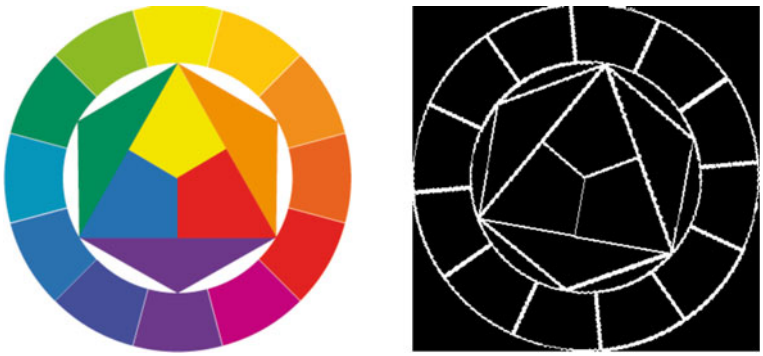

Fig. 1 Segmentation result of a 340×340 pixels image computed in 0.105 s

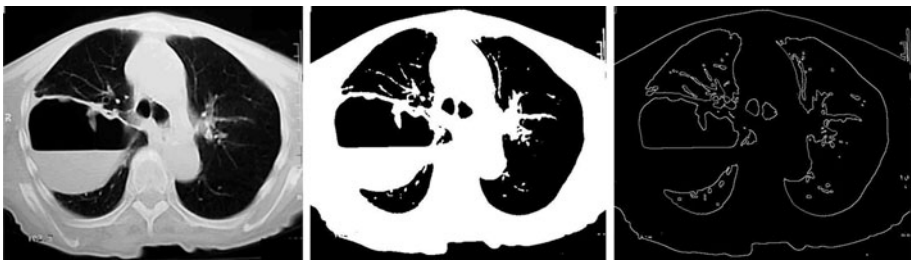

Fig. 2 Segmentation result of a 600×600 CT image of human lungs, computed in 0.33 s. On the *left*, the initial image, on the *middle* the binarized image, and on the *right* the segmentation result

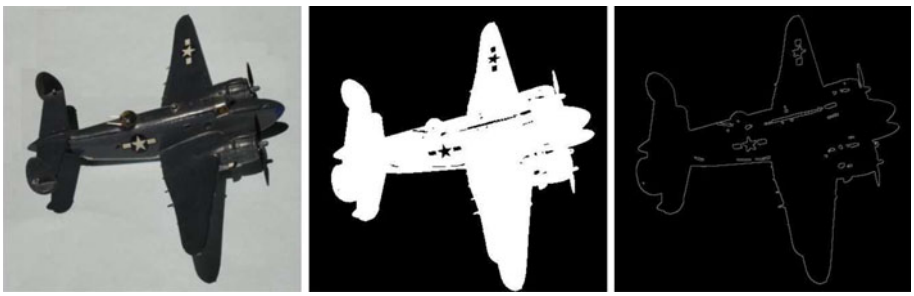

Fig. 3 Segmentation result of a 437 × 437 image, computed in 0.31 s. On the *left*, the initial image, on the *middle* the binarized image, and on the *right* the segmentation result

shown. It is a 600 × 600 medical image that corresponds to computed tomography (CT) human lungs. In that case, the software tool needs to preprocess the initial image before segment it. This fact is due to the simplicity of the segmentation algorithm implemented here, and future improvements of it are planned. In this case, the preprocessing transforms an image with many gray levels into a black and white image (middle image in Fig. 2). The output of the system is show on the right picture. The segmentation process took 0.33 s. A similar example can be seen in Fig. 3. In this case, the image of a toy of size 437 × 437 was segmented in 0.31 s.

5 Conclusions and future work

Segmentation has features which makes it suitable for techniques from Natural Computing. Local solution or the parallelism of the process can be studied from a theoretical point of view, but for an effective application of these techniques to the real world, it is necessary to have an appropriate software tool.

This paper represents an improvement with respect to the simulator presented in Borrego-Ropero et al. (2007). Our software tool is able to deal with images of reasonable size and can become a helping tool for the treatment of digital images, as the two last examples show.

The software can be improved and several research lines are open. One of them is to study the influence of the type of adjacency (4 or 8) on the result. The possibility of including preprocessing features in our tool be also studied. As a final remark, we will consider to adapt this software to a parallel hardware architecture and exploit in a realistic way the intrinsic parallelism of membrane computing methods.

Acknowledgements DDP and MAGN acknowledge the support of the projects TIN2008-04487-E and TIN-2009-13192 of the Ministerio de Ciencia e Innovación of Spain and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200. PR and HMA acknowledge the support of the project MTM2009-12716 of the Ministerio español de Educación y Ciencia, the project PO6-TIC-02268 of Excellence of Junta de Andalucía, and the "Computational Topology and Applied Mathematics" PAICYT research group FQM-296.

References

Abelson H, Allen D, Coore D, Hanson C, Homsy G Jr, Knight TF, Nagpal R, Rauch E, Sussman GJ, Weiss R (2000) Amorphous computing. Commun ACM 43(5):74–82

Adleman LM (1994) Molecular computation of solutions to combinatorial problems. Science 266:1021–1024

Borrego-Ropero R, Díaz-Pernil D, Pérez-Jiménez MJ (2007) Tissue simulator: a graphical tool for tissue P systems. In: Vaszil G (ed) Proceedings of the international workshop automata for cellular and molecular computing. Satellite of the 16th international symposium on fundamentals of computational theory. MTA SZTAKI, Budapest, Hungary, pp 23–34

Ceterchi R, Gramatovici R, Jonoska N, Subramanian KG (2003) Tissue-like P systems with active membranes for picture generation. Fundam Inform 56(4):311–328

Ceterchi R, Mutyam M, Păun G, Subramanian KG (2003) Array-rewriting P systems. Nat Comput 2(3):229–249

Chao J, Nakayama J (1996) Cubical singular simplex model for 3D objects and fast computation of homology groups. In: 13th International conference on pattern recognition (ICPR'96), vol IV. IEEE Computer Society, Los Alamitos, CA, USA, pp 190–194

Christinal HA, Díaz-Pernil D, Real P (2009a) Segmentation in 2D and 3D image using tissue-like P system. In: Bayro-Corrochano E, Eklundh JO (eds) CIARP, lecture notes in computer science, vol 5856. Springer, Berlin, pp 169–176

Christinal HA, Díaz-Pernil D, Real P (2009b) Using membrane computing for obtaining homology groups of binary 2D digital images. In: Wiederhold P, Barneva RP (eds) IWCIA, lecture notes in computer science, vol 5852. Springer, Berlin, pp 383–396

Christinal HA, Díaz-Pernil D, Real P (2010) P systems and computational algebraic topology. J Math Comput Model 52(11–12):1982–1996. The BIC-TA 2009 special issue, international conference on bio-inspired computing: theory and applications

de Castro LN, Timmis J (2002) Artificial immune systems: a new computational intelligence approach. Springer, Heidelberg

Díaz-Pernil D, Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Riscos-Núñez A (2008a) A uniform family of tissue P systems with cell division solving 3-COL in a linear time. Theor Comput Sci 404(1–2):76–87

Díaz-Pernil D, Pérez-Hurtado I, Pérez-Jiménez MJ, Riscos-Núñez A (2008b) A P-lingua programming environment for membrane computing. In: Corne DW, Frisco P, Păun G, Rozenberg G, Salomaa A (eds) Workshop on membrane computing, lecture notes in computer science, vol 5391. Springer, Berlin, pp 187–203

Díaz-Pernil D, Pérez-Jiménez MJ, Romero A (2009) Efficient simulation of tissue-like P systems by transition cell-like P systems. Nat Comput 8:797–806

Díaz-Pernil D, Graciani C, Gutiérrez-Naranjo MA, Pérez-Hurtado I, Mario J. Pérez-Jiménez M (2010) Software for P systems. In:

Păun G, Rozenberg G, Salomaa A (eds) The Oxford handbook of membrane computing. Oxford University Press, Oxford, pp 437–454

Engelbrecht AP (2005) Fundamentals of computational swarm intelligence. Wiley, Chichester

Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Riscos-Núñez A (2006) Available membrane computing software. In: Ciobanu G, Pérez-Jiménez MJ, Păun G (eds) Applications of membrane computing, natural computing series. Springer, Berlin, pp 411–436

Holland JH (1992) Adaptation in natural and artificial systems. MIT Press, Cambridge

Kari L, Rozenberg G (2008) The many facets of natural computing. Commun ACM 51(10):72–83

Kim SH, Kim HG, Tchah KH (1998) Object oriented face detection using colour transformation and range segmentation. Electron Lett IEEE 34:979–980

Kropatsch WG, Haxhimusa Y, Ion A (2007) Multiresolution image segmentations in graph pyramids. In: Kandel A, Bunke H, Last M (eds) Applied graph theory in computer vision and pattern recognition, studies in computational intelligence, vol 52. Springer, New York, pp 3–41

Martín-Vide C, Pazos J, Păun G, Rodríguez-Patón A (2002) A new class of symbolic abstract neural nets: Tissue P systems. In: Ibarra OH, Zhang L (eds) COCOON, lecture notes in computer science, vol 2387. Springer, Berlin, pp 290–299

Martín-Vide C, Păun G, Pazos J, Rodríguez-Patón A (2003) Tissue P systems. Theor Comput Sci 296(2):295–326

McCulloch WS, Pitts W (1988) A logical calculus of the ideas immanent in nervous activity. Neurocomputing: foundations of research. pp 15–27. Originally Published in McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5:115–133

P system web page. http://ppage.psystems.eu

Păun G (2000) Computing with membranes. J Comput Syst Sci 61(1):108–143. doi:10.1006/jcss.1999.1693. See also Păun G (1998) Computing with membranes. Tech. Rep. 208. Turku Centre for Computer Science, Turku

Păun G (2002) Membrane computing. An introduction. Springer-Verlag, Berlin

Păun G, Rozenberg G, Salomaa A (eds) (2010) The Oxford handbook of membrane computing. Oxford University Press, New York

Rosenfeld A (1970) Connectivity in digital pictures. J Associ Comput Mach 17(1):146–160. doi:10.1145/321556.321570

Rosenfeld A (1979) Digital topology. Am Math Mon 86(8):621–630

Shapiro LG, Stockman GC (2001) Computer vision. Prentice Hall PTR, Upper Saddle River

Tobias OJ, Seara R (2002) Image segmentation by histogram thresholding using fuzzy sets. IEEE Trans Image Process 11(12): 1457–1465

von Neumann J (1966) In: Burks AW (ed) Theory of self-reproducing automata. University of Illinois Press, Champign, USA

Wang D, Lu H, Zhang J, Liang JZ (2005) A knowledge-based fuzzy clustering method with adaptation penalty for bone segmentation of ct images. In: Proceedings of the 2005 IEEE engineering in medicine and biology 27th annual conference, vol 6. pp 6488–6491

Yazid H, Arof H (2008) Image segmentation using watershed transformation for facial expression recognition. In: IFMBE proceedings, 4th Kuala Lumpur international conference on biomedical engineering, pp 575–578

Yuan X, Situ N, Zouridakis G (2009) A narrow band graph partitioning method for skin lesion segmentation. Pattern Recogn 42(6):1017–1028. doi:10.1016/j.patcog.2008.09.006