Content-Based Image Recognition using Cellular Discrete-Event System Specifications Methodology

Mohammad Moallemi, Gabriel Wainer
Dept. of Systems and Computer Engineering
Carleton University, Centre of Visualization and Simulation (V-SIM)
{moallemi, gwainer}@sce.carleton.ca

Keywords: Cell-DEVS, Real-Time, Content-Based, Image-Recognition.

Abstract
Context-aware application development for mobile systems is a new trend in ubiquitous computing systems research. The idea is to capitalize on contextual data (i.e. user location, time of day, nearby facilities and people, user activity, etc.) in order to satisfy user-specific needs and offering relevant data and services to the audience. Here, we show hoe to use the Cell-DEVS methodology to apply intelligent object recognition algorithms to solve the above research questions. The cellular nature of the modeling approach and the rule-base behavior definition for cells provides a platform for pixel-wise operations, leading to easier and faster adoption and implementation of content-based algorithms. The other advantage of this method is its fast computing apparatus working asynchronously on the cellular grid, increasing the execution speed.

1. INTRODUCTION

The overwhelming adoption of digital images over the past decade has raised a significant need for efficient content-based access techniques to these images. The large number of images stored in digital databases has made it difficult to search a specific image or a pattern in an image. The advantage of searching images based on meta-data is faster and more accurate results; however documenting large number of images requires significant human effort in every image and is almost impractical. Content-Based Image Retrieval (CBIR) or query by image content allows digital devices to retrieve information about the image based on the image characteristics such as color, shape, texture and any other textual data. This technique can be used in solving problems and facilitating workflow in art collections, photograph archives, medical diagnosis, retail catalogues, crime prevention, intellectual property, engineering design, geographical information systems and many more [1].

On the other hand, the wide-spread usage of digital cameras, cell-phones, tablets and mobile computing devices with built-in cameras has escalated the need for real-time image recognition techniques. Context-aware application development for mobile and web-based systems is a new trend in mobile and ubiquitous computing systems research field. These techniques have attracted significant attention in the past decade with the fast growing use of mobile computing devices. These mobile applications can capitalize on contextual data (i.e. user location, time of day, nearby facilities and people, user activity, etc.) in order to satisfy user-specific needs and offering relevant data and services.

One of the context aware cross-platform applications is the inSitu Solutions™ [2], working in Web and mobile phones and tablet platforms that allows publishing location-based content. The objective of such application is informing, attracting and retaining visitors in public places (such as museums, parks, natural reserves, tourist sites, universities and schools, conferences and events, shopping centers, airports, train stations, subways, etc.).

The research question to be addressed in this project is how to recognize one or several pre-specified or learned hotspots or object classes, and then identifies individual instances of them using the inSitu Solutions. Once specific hotspots are detected, they should link to the source of information regarding that specific hotspot. The challenges are in detecting hotspots in complex artworks and artifact images, considering the large number of hotspot instances, and the complexity of the interpretation of content, when numerous objects are found in the nearby location. These circumstances demand innovative and efficient image processing, interactive media, and visitor behavior-analysis techniques specific to the mobile embedded environments. The idea is thus to allow the visitors to be immersed in the application, allowing them to be aware of their environment at a specific point of interest. Then, the visitor should be able to access very specific information, when looking at the environment through the device's camera.

The objective is to use an abstract model to recognize one or several pre-specified or learned hot-spots or object classes, and then identify individual instances of the recognized objects in the scenery images. The idea is to achieve this by using a theoretical approach (in particular, discrete-event methodologies), resolving the image processing problems using the Cell-DEVS (Cellular-Discrete EVent System Specification) formalism [18]. Cell-DEVS is a combination of Cellular Automata and DEVS [19] with explicit timing delays, which solves the problem of unnecessary processing burden in cells and allows for more efficient asynchronous execution, using a continuous time base, without losing accuracy. Cell-DEVS was originally introduced for Modeling and Simulation (M&S) of spatial systems however, there is

a potential in using the method for image processing and hotspot detection. The solution will be implemented on CD++ [18] a M&S tool that provides a development environment for implementing DEVS and Cell-DEVS models. CD++ provides a C++ programming environment in which DEVS and Cell-DEVS models can be developed and incorporated into the simulator class hierarchy. Finally the CD++ simulator along with the executive model will be embedded on the mobile platform interoperating with the inSitu solutions.

We discuss different problems in designing and developing such an augmented reality component, and we propose and innovative design solution for this kind of application on smartphones (using the iPhone OS (iOS) and Google Android platforms). The presented theory and methodology will be deployed in inSitu Solutions and upgrade the product's features, capabilities and performance.

2. RELATED WORK

Early digital content-based search algorithms were based on computer vision, focusing on the similarity of digital images, audio and video [3][4]. Internet-based image content search engines such as Webseek [5] and Webseer [6] evolved shortly after the initial attempts which were integrated with large enterprise digital databases.

Content-based visual search began with text-based query of the image content. One of the commercial tools for CBIR is IBM's QBIC (Query By Image Content) [7]. QBIC allows image searching using any combination of text, color, shape and texture (http://wwwqbic.almaden.ibm.com). The search is done based on query-by- example, specifying a sample image and looking for similar images. A palate image or a drawn sketch can also be fed to the system. The images are indexed using R*Tree content-based ing [8]. The system matches the query with the images in the database, calculating a similarity score and returning the highest scored images.

Another commercial tool for CBIR is VIRAGE [9]. VIRAGE is library available as program modules that can be integrated with the software under development. Like QBIC, VIRAGE makes use of the color features of the query image to find similar images stored in the library. It can also be added to available database management systems such as Oracle or Informix as an add-on. AltaVista photo search uses VIRAGE to query the images online.

There is varied research on CBIR in the literature, each of them applied to specific applications, as surveyed in [10]. For instance, Cortina [11] is a CBIR search engine working on the WWW, matching query-by-image and query-by-keyword searches. It compares low-level features (color, texture, edges) with 11 million images in the database. Simplicity [12] is composed of a proprietary dataset of 60000 objects. The search can be done based on several querying options, such as random proposition of objects, a drawing provided by the user, and selection of items displayed by the system. Tiltomo [13] is capable of querying images on the flicker online repository. The search is based on keywords (by comparing keywords with user-provided image annotations) and image content. The CBIR is based on low-level similarity matching and textual comparison. The content-based search tool efficiently compares images texture and color and extracts similar images.

The following research works are more closely related to our work. MIDP [14] is simple CBIR system implemented on Nokia mobile phones, querying an example image with the images stored in the phone's memory using a client- server scheme. The system compares similarity in histogram, texture and color, and it returns a score for each image. The main processing is done on the server side, due to low processing power on the mobile devices and the slow wireless communication speed. A similar attempt was done in [15], with the same client server approach. Here, the system performance improved; however the system is still prone to the same limiting factors (hardware, wireless links). The indexing of the multimedia content used in these approaches is described in [16]. Other similar works have been published by the same authors. The common deficiency in these systems is the lengthy response time of the queries, due to the limitations of hardware and communications.

In this work, we will try to analyze the use of theoretical approaches (in particular, discrete-event methodologies) in order to resolve these image processing problems. In particular, Cellular automata (CA) [17] has been used with this purpose [24][25]. CA theory represents a model as a cellular grid, in which each cell is a state machine. The time advances in a discrete manner, triggering state changes in the cells, based on the value of their neighbor cells. CA has been used for experimental models, but it was not deployed in an industrial scale. Cell-DEVS (Cellular Discrete Event System Specification) formalism [18] is an improved derivative of CA, which solves the problem of unnecessary processing burden in cells and allows for more efficient asynchronous execution, using a continuous time base, without losing accuracy. In this methodology, each cell is represented as a DEVS atomic model that changes state in response to the occurrence of events in an event-driven fashion. Cell-DEVS was originally introduced for M&S of spatial systems; here, we want to explore its adoption for modeling image processing applications, CBIR or hotspot detection.

In this research, we propose using Cell-DEVS methodology to apply intelligent object recognition algorithms (e.g. Edge matching, Gradient matching, Pose consistency, Geometric hashing, etc.). The cellular nature of the Cell-DEVS modeling approach and the rule-base behavior definition for cells provides a platform for pixel-wise operation definition,

leading to easier and faster adoption and implementation of POI (Point Of Interest) detection and CBIR algorithms. The other advantage of this method is its fast computing apparatus working asynchronously on the cellular grid, increasing the execution speed. The formalism also allows for straightforward modeling of CBIR and multi-image algorithms. Finally, the formal I/O port definitions in the formalism permits producing output signals based on specific condition satisfaction in the cell lattice, allowing for data transfer between different spatial components. The idea is to convert images and sceneries to 2D cellular models, apply the proposed algorithms, and finally integrate them with the inSitu™ multimedia solutions.

Cell-DEVS [18] is an extension to DEVS [19] that allows defining cellular models with explicit timing delays. A Cell-DEVS model is a lattice of cells holding state variables and a computing apparatus, which is in charge of updating the cell states according to a local rule. This is done using the current cell state and those of a finite set of nearby cells (called its neighborhood). Cell-DEVS improves execution performance of cellular models by using a discrete-event approach. It also enhances the cell's timing definition by making it more expressive. Each cell is defined as a DEVS atomic model, and it can be later integrated to a coupled model representing the cell space. Cell-DEVS are informally defined as shown in Figure 1.

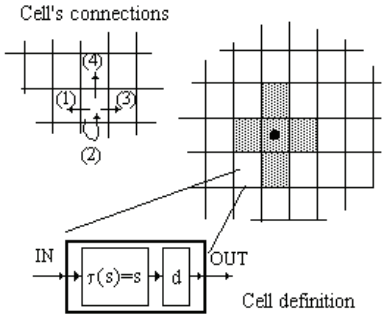

Figure 1. Cell-DEVS model.

Each cell uses N inputs to compute its next state. These inputs, which are received through the model's interface, activate a local computing function (τ). A delay (d) can be associated with each cell. The state (s) changes can be transmitted to other models, but only after the consumption of this delay. Once the cell behavior is defined, a coupled Cell-DEVS can be created by putting together a number of cells interconnected by a neighborhood relationship.

CD++ [18] is a M&S tool that provides a development environment for implementing DEVS and Cell-DEVS models. DEVS Atomic models can be developed and incorporated into a class hierarchy programmed in C++. Coupled models can be defined using a built-in specification language. Cell-DEVS models are built following the formal specifications for DEVS, and a built-in language is provided to describe the behavior rules. The language is based on the formal specifications of Cell-DEVS. The model specification includes the definition of the size and dimension of the cell space, the shape of the neighborhood and borders. The cell's local computing function is defined using a set of rules with the form POSTCONDITION DELAY {PRECONDITION}. These indicate that when the PRECONDITION is satisfied, the state of the cell will change to the designated POSTCONDITION, whose computed values will be transmitted to other components after consuming the DELAY. If the precondition is false, the next rule in the list is evaluated until a rule is satisfied or there are no more rules.

3. HOTSPOT DETECTION USING CELL-DEVS

As described in the Introduction, the main challenge is to detect pre-specified hotspots in images (for instance, artwork in museums, pictures in Zoos) while the user is scanning the image using a cell-phone camera. The system must process the image coming from the camera, compare its contents with the entire image, and try to detect the specific location in the entire image where the camera is pointing to. This process must be done in real-time, and must be fast enough so that the user can see a constant streaming video. As soon as the camera gaze vector is located in the image, information about the hotspots available in that frame can be extracted from the database and rendered to the user.

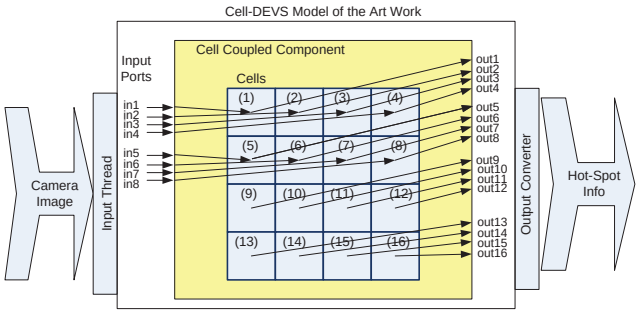

Figure 2. Image cell space with I/O ports

Figure 2 shows the cell space we will use, including its input and output ports. The input ports carry image pixels coming from the camera and they are stored in a first series of cells. These values will not affect the cell values and are stored in the input queue of each cell, waiting for the similarity matching function to access them.

In order to solve this challenge, we used Cell-DEVS modeling, and defined content-based comparison of the two images. The following steps summarize the solution:

1) First, the user location is found using GPS or RFID technology, to detect which image s/he is looking at.
2) A digital copy of the image is downloaded to the user's mobile device, and a Cell-DEVS model (C) is built with the dimensions of the image. Each cell in the cell space

represents a pixel, and the value of the cell is the RGB color code associated with that pixel.

3) A neighborhood N equal to the size of the real-time image from device's camera is created for this cell-space. The neighborhood is used to verify the similarity of the camera image with the same size frame (neighborhood) in the entire image. In this way, the camera image is compared with the image in one timestep.
4) An external input port set X is established containing *k* input ports equal to the size of the camera image (or the neighborhood). These ports are used during the execution to receive camera image periodically and perform the similarity matching.
5) An external output port set Y is created containing *l* ports equal to the size of the cell space (image size). In other words, each cell owns an output port to output the similarity score of its neighborhood with the camera image at each time step.
6) The rule-base implementation defines the similarity matching algorithm, which scores the similarity of each cell's neighborhood with the camera image. This process continues periodically, ensuring the real-time functionality of the system.

3.1. Challenges

A number of challenges must be tackled in this research. Several factors can affect the camera image such as shading, fading, lighting, resolution, gaze vector etc. There are different similarity factors used by similarity functions that can be used. The Gaussian Mixture Vector Quantization (GMVQ) [20] is used to extract color histograms from the images and compare the results. This method has shown promising results and eliminates some of the above mentioned effects. On the other hand, shape similarity measure using discrete curve evolution [21] is also useful to wave the noise in the shape. Shape matching can also be done using shape descriptors [22] which provides a compact and robust solution to this problem and supports geometric transformations. A dynamic programming (DP) approach to shape matching has been proposed in [23]. The size of the cell-space neighborhood can also be a challenge. The proportion of the camera image to the image depends on the distance between the camera and the image. As the camera distances from the image, the camera image covers a bigger frame of the image, challenging the size of the neighborhood. Shape matching algorithms might be better candidates for the above mentioned challenges rather than color similarity functions, as the former is more independent of the dimensions of the images.

Another challenge is the performance and probability of error in the pattern matching. Because of the real-time nature of the system and also limited processing resources on the mobile device, it is necessary to choose an efficient approach. Shape matching approaches are prone to high computation pits and might not outperform color matching approaches [10]. We believe that Cell-DEVS theory and its fast computing apparatus will outperform the current image processing approaches and provides a straight forward platform for implementing similarity functions.

4. PRELIMINARY IMPLEMENTATION

A preliminary implementation of the proposed cellular content-based image matching has been defined using CD++. A sample image is modeled as a Cell-DEVS grid. A series of input events have been generated from this image and fed to the model representing the camera image frame. A rule-base algorithm has been defined to match the input image with the neighborhood in each cell. Whenever all of the neighbor cells of the cell being evaluated match the camera image coming from the input ports, the central cell produces an output. The output port associated with the central cell in the matching neighborhood indicates the coordinate of the center of the frame where the camera is gazing at.

Figure 3 shows the example image used for content based matching. The image is used to create a Cell-DEVS model with dimensions of 86 by 49 cells (pixels). The camera image resolution is set to 7 by 7 pixels. Thus, a Moore neighborhood with the same size has been defined. 49 input ports have been added to the model to carry the camera image periodically. Each cell has an output port to signal the matching, thus there are 4214 output ports.

Figure 3. Example image

The formal specification of the Cell-DEVS model for the proposed image is given bellow:

M=<*I*, *X*, *Y*, *Xlist*, *Ylist*, *n*, *N*, *{n1,n2}*, *C*, *B*, *Z*, *select*>

Where:

I= <PX,Py>, with PX={in1, in2 … in49}, Py={output3, output4 … output4216};

X=*Y*= {0 … 16777216};

Xlist= {in1 → (0,0), … in49 →(0,48)}

Ylist= {(0,0) → output3, … (85,48) → output4216}

n=49

N= see Figure 4

{n1, n2 }= {86,49}
C={Cij / i∈[1,86], j∈[1,49]}
B= nowrapped;

Figure 4 shows the neighborhood used in this model, with the relative coordinates from the central cell. 49 cells are involved in the neighborhood and are evaluated for similarity with the camera image. This process occurs for each cell in the cell-space.

(-3,-3)	(-3,-2)	(-3,-1)	(-3,0)	(-3,1)	(-3,2)	(-3,3)
(-2,-3)	(-2,-2)	(-2,-1)	(-2,0)	(-2,1)	(-2,2)	(-2,3)
(-1,-3)	(-1,-2)	(-1,-1)	(-1,0)	(-1,1)	(-1,2)	(-1,3)
(0,-3)	(0,-2)	(0,-1)	(0,0)	(0,1)	(0,2)	(0,3)
(1,-3)	(1,-2)	(1,-1)	(1,0)	(1,1)	(1,2)	(1,3)
(2,-3)	(2,-2)	(2,-1)	(2,0)	(2,1)	(2,2)	(2,3)
(3,-3)	(3,-2)	(3,-1)	(3,0)	(3,1)	(3,2)	(3,3)

Figure 4- Moore neighborhood for the model

This model was implemented on CD++. In order to implement this model and prepare a basis for adding similarity matching and mobile embedded platform compatibility, the source of CD++ code was modified. In order to avoid the manual declaration of input/output ports, they are automatically generated for each cell. A new cell type, named "image" has been added to CD++, whose value do not change over time, and also holds input values in a queue which does not alter the cell value. This allows for using DEVS input ports for injecting periodic inputs from the camera, and storing them in the cells without affecting the cell values. A new function "cellPortValue(x,y)" has been added to the CD++ model file parser, allowing a modeler to access an available input anywhere in the cell space. This function provides access to camera image pixels usable for similarity matching rules. So far, the model is composed of simple rules that compare the values of each cell in the neighborhood with the value of the associated input port.

To verify the initial implementation, we show two series of inputs injected to the model via CD++ event file. Each input covers a frame of 7 by 7 pixels in the image, where the camera is pointing at. Figure 5 shows the image using a resolution of 86 by 49 pixels, in which the two input frames are indicated. The color values of the containing pixels are serialized in the event file and injected to the model with a period of 1 millisecond.

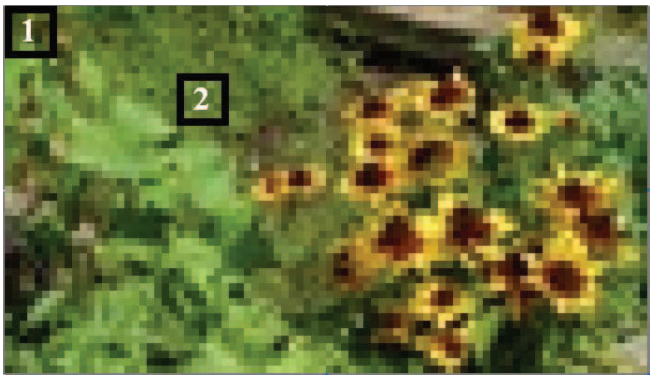

Figure 5. Input Frames in the image

The following code snippet illustrates a part of the image matching rule in CD++. As it is observed, line 2 produces an output of 1 (after 1 ms), if the condition specified within line 3 to 7 is satisfied. The condition verifies the values of the input ports with the values in the neighborhood cells, in a serial sequence for all of the 49 cells. This rule is evaluated for each cell in the entire cell-space (the image). If all of the cell's neighbors match the input frame, the central cell outputs value 1, indicating a match.

```
1   [rules]
2   rule : { (0,0)+ send(output,1) } 1{
3   cellPortValue(0,0) = (-3,-3) and
4   cellPortValue(0,1) = (-3,-2) and
5   ...
6   cellPortValue(0,47) = (3,2) and
7   cellPortValue(0,48) = (3,3)}
8   rule : { (0,0) } 1 { t }
```

The following box shows a sample output produced during the execution of the model, corresponding to the two series of input frames. The first line shows the first output produced after 1 millisecond from cell 264 which has the coordinate (4,4), indicating the center of the frame 1 in Figure 5. The second line is produced 2 milliseconds after the start of the execution (when the second input series has been injected). The cell 2163 corresponding to coordinate (25,10) has produced the output value 1, which is the central cell in the frame 2. The test model recognized the image frame (camera image) in the image. The next step will be integrating advanced similarity functions with this basis and embedding it in mobile environments.

```
1   00:00:00:001 output264 1
2   00:00:00:002 output2163 1
```

5. CONCLUSIONS AND FUTURE WORK

The goal of this research is to explore use Cell-DEVS theory in content-based image recognition in context-aware applications in mobile devices. The research problem is how to recognize one or several pre-specified or learned hotspots or object classes, and then identify individual instanc-

es of the recognized objects in the scenery images. The cellular nature of the Cell-DEVS modeling approach and the rule-base behavior definition for cells provides a platform for pixel-wise operations, leading to easier and faster adoption and implementation of content-based algorithms. In addition, its fast and optimized computing apparatus helps us solve the performance and latency problem in similar technologies. A preliminary example model has been developed in CD++.

The research is a work in progress, and future steps will include refinement of CD++ initialization steps to speed up this process, integrating flattened coordinator technique to reduce the message passing burden between the modeling levels, incorporating advance similarity functions in the rule-base to overcome the challenges mentioned in section 3.1, and finally integrating the final solution with inSitu™ solutions and embedding it in the mobile operating systems such as iOS and Android.

6. ACKNOWLEDGEMENTS

This Research was funded by NSERC/ENGAGE grant to G. A. W. and Kanvasys Corporations. The authors wish to thank Jean-Baptiste Minchelli Director of Technology at Idéeclic for his excellent supervision and support in this research.

7. REFERENCES

[1] Yong Rui, Thomas S. Huang, and Shih-Fu Chang, “Image Retrieval: Current Techniques, Promising Directions and Open Issues”, Journal of Visual Communication and Image Representation, Vol. 10, pp. 39-62, March, 1999.

[2] inSitu Solutions webpage on Kanvasys Website available at www.kanvasys.net/node/55. Accessed Nov. 2011.

[3] Ballard, D.H. and Brown, C.M.. “Computer Vision”. Prentice Hall, New Jersey, USA., 1982.

[4] Levine, M. “Vision in Man and Machine”, Mcgraw Hill, Columbus, 1985.

[5] Smith, J. R. and Chang, S.F. “Visually Searching the Web for Content”, Journal of IEEE Multimedia 4(3), 12-20. 1977.

[6] Frankel, C., Swain, M.J., and Athitsos, V. “WebSeer: An Image Search Engine for the World Wide Web”, University of Chicago Technical Report 96-14, University of Chicago, USA. 1996.

[7] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Qian Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, “Query by image and video content: the QBIC system” IEEE Computer Vol. 28, No 9, pp. 23-32. 1995.

[8] C. Faloutsos , W. Equitz , M. Flickner , W. Niblack , D. Petkovic , R. Barber, “Efficient and effective querying by image content” Journal of Intelligent Information Systems Vol. 3, pp. 231-262. 1994.

[9] J. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, “The Virage image search engine: An open framework for image management. In: Storage and Retrieval for Image and Video Databases. Proceedings of SPIE Journal. Vol. 2670, pp. 76–87, 1996.

[10] R. Datta, J. Li, and J. Z. Wang, “Content-based image retrieval - Approaches and trends of the new age,” In Proc. Int. Workshop on Multimedia Information Retrieval, pp. 253–262, 2005.

[11] T. Quack, U. Monich, L. Thiele, and B. S. Manjunath, “Cortina: A System for Largescale, Content-based Web Image Retrieval” In Proc. of 12th ACM Int. Conf. on Multimedia, New York, NY, USA, 2004.

[12] J. Li J. Wang and G. Wiedergold, “SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries”. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 23, No. 9, pp. 947 – 963, September 2001.

[13] Tiltomo website: http://www.tiltomo.com (accessed on January 23, 2007)

[14] A. Iftikhar, F. Alaya Cheikh, B. Cramariuc and M. Gabbouj, “Query by Image Content using NOKIA 9210 Communicator” , Proc. of the Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS'01, pp.133-137, Tampere, Finland, May 2001.

[15] I. Ahmad, S. Abdullah, S. Kiranyaz, M. Gabbouj, “Content-based image retrieval on mobile devices”, in Proceedings of SPIE (Multimedia on Mobile Devices), Vol. 5684, San Jose, CA, USA, January 2005.

[16] O. Guldogan, M. Gabbouj, “Content-based image indexing and retrieval framework on symbian based mobile platform”, Proc. European Signal Processing Conference, EUSIPCO, Antalya, Turkey, Sep. 2005.

[17] Wolfram, S. "Theory and applications of cellular automata". Vol. 1. Advances Series on Complex Systems. World Scientific. Singapore. 1986.

[18] G. A. Wainer, "Discrete-event modeling and simulation; a practitioner's approach", CRC / Taylor & Francis, ISBN: 9781420053364, 2009.

[19] B. Zeigler, T. Kim, H. Praehofer, “Theory of Modeling and Simulation”. Academic Press. 2000.

[20] S. Jeong, C. S. Won, and R.M. Gray, “Image retrieval using color histograms generated by Gauss mixture vector quantization,” Computer Vision and Image Understanding, 9(1–3):44–66, 2004.

[21] L.J. Latecki and R. Lakamper, “Shape Similarity Measure Based on Correspondence of Visual Parts,” IEEE Trans. Pattern Analysis and Machine Intelligence,22(10):1185–1190, 2000.

[22] S. Belongie, J. Malik, and J. Puzicha, “Shape Matching and Object Recognition Using Shape Contexts,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 24, No. 4, pp. 509–522, 2002.

[23] E. G. M. Petrakis, A. Diplaros, and E. Milios, “Matching and Retrieval of Distorted and Occluded Shapes Using Dynamic Programming,” IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 24, No. 4, pp. 509–522, 2002.

[24] B. Viher, A. Dobnikar, and D. Zazula, “Cellular automata and follicle recognition problem and possibilities of using cellular automata for image recognition purposes,” Int. J. Med. Inf., vol. 49, no. 2, pp.231–241, 1998.

[25] Paul L Rosin, Training Cellular Automata for Image Processing, IEEE Transactions on Image Processing, Vol 15, No. 7, pp. 2076 – 2087, July 2006