High Performance VLSI Architecture for Data Clustering Targeted at Computer Vision

Orlando J. Hernandez
Department Electrical and Computer Engineering
The College of New Jersey
hernande@tcnj.edu

Abstract

This paper presents a high performance architecture for the important task of unsupervised data clustering in computer vision applications. This architecture is suitable for VLSI implementation, as it exploits paradigms of massive connectivity like those inspired by neural networks, and parallelism and functionality integration that can be afforded by emerging nanometer semiconductor technologies. By utilizing a "global-systolic, local-hyper-connected" architectural approach, this architecture can be suitable for the processing of real time DVD quality video at the highest rate allowed by the MPEG-2 standard. This implies a performance improvement of 118 times or better than approaches using conventional compute platforms.

1. Introduction

As new algorithms are developed using a paradigm of off-line non real-time implementation, many times there is a need to adapt and advance the state of the art of hardware architectures to implement such algorithms in a real-time manner if they are to truly serve a useful purpose in industry and defense, and beyond an academic setting. Such is the case with many underlying algorithms used in computer vision paradigms. Specifically, of interest are high speed hardware architectures for the implementation of real time unsupervised data clustering.

This paper addresses the mapping of the unsupervised histogram peak-climbing clustering algorithm to a novel high speed architecture suitable for VLSI implementation and real-time performance. Specifically, this architecture exploits paradigms of massive connectivity like those inspired by neural networks, and parallelism and functionality integration that can be afforded by emerging nanometer semiconductor technologies. Special attention is paid to the clustering of high dimensionality sparse data sets like those found in the clustering of information rich features used for color image segmentation and computer vision, and "orders of magnitude" performance increase from current implementation on a generic compute platform.

These architectures will aide computer vision technology to deliver on its promise of real-time data processing and information generation, and these are solutions of special interest to industry and defense. An example of applications to defense is the automatic analysis of scenes for the recognition of targets and foes in battlefields.

Clustering algorithms can be implemented in conventional compute platforms, but while these can have a high degree of flexibility, they do carry the burden of not being tuned specifically for the task of clustering, and therefore suffer from a high amount of overhead and are inherently inefficient. For instance, the clustering algorithm used in this work has been benchmarked as requiring 172 mS running on a 2.27 GHz processor with virtually infinite memory (RAM – not disk) to execute the algorithm. This instance of the algorithm was clustering 961 vectors of 22 dimensions each, which is equivalent to an image resolution of 128 x 128 pixels. This, by no means, comes close to the levels of performance necessary for real-time video processing and higher resolutions.

2. Some recent related work

Significant advances in the quality of color image segmentation results have recently been reported in the literature [1], [2]. This methodology uses high dimensionality Multispectral Random Field Texture Models [3], [4] and Color Content as features of a sub-image defined by a sliding window. These features are in turn clustered using an unsupervised peek-climbing algorithm in the highly multidimensional feature space. Once the features are clustered, these clusters are mapped back to the spatial domain of the image which results in the image segmentation.

Although some highly talented researchers have devoted great efforts to devising hardware architectures to accelerate the execution of clustering algorithms, these efforts have not fully addressed the high performance

0-7803-8865-8/05/$20.00 ©2005 IEEE.

demands of real-time high quality color video processing [5], [6], [7], [8], [9], [10]. The specific problem of conceiving architectures for the very efficient unsupervised peak- climbing clustering algorithm has not been addressed either. Some of the literature refers to the same type of architectural approach [6], [7], [8], [9] while other efforts have been mostly focused on aiding the performance of Artificial Neural Networks (ANNs) [11], [12], [14], [15], [16], [17], [18] or apply to specific problem domains [19], and many of the architectures reported have been of an analog nature [11], [12], [13], [20]. While analog processing tends to necessitate fewer components for a given complex operation, the technology and the approach suffer from several drawbacks. Namely, analog VLSI technology is more expensive to manufacture and test, is less accurate than a digital implementation, suffers from serious sensitivity to noise, and its performance is very susceptible to changes in supply voltage and environmental temperature conditions. Compensating for all these susceptibilities and obtaining a robust design may require added complexity, which may hinder the reliable manufacturability of the circuitry.

3. Clustering algorithm

This section describes the clustering algorithm implemented in this work. Given M features **f** of dimensionality N to be clustered, the first step is to generate a histogram of N dimensions [21]. This histogram is generated by quantizing each dimension according to the following equations:

$$CS(k) = \frac{f_{max}(k) - f_{min}(k)}{Q} \quad ; \quad k = 1, 2, \ldots, N \qquad (1)$$

$$d_k = INT\left\{\frac{f(k) - f_{min}(k)}{CS(k)} + 1\right\} \quad ; \quad k = 1, 2, \ldots, N \qquad (2)$$

for each of the M f(k) feature members, where:

N = dimensions of the features

CS(k) = length of the histogram cell in the kth dimension

$f_{max}(k)$ = maximum value of the kth dimension of the M features

$f_{min}(k)$ = minimum value of the kth dimension of the M features

Q = total number of quantization levels

d_k = index for a histogram cell in the kth dimension associated with a given feature **f**

Since the dynamic range of the vectors in each dimension can be quite different, the cell size for each dimension would be different. Hence the cells will be hyper-boxes. This provides efficient dynamic range management of the data, which will tend to enhance the quality and accuracy of the results. Next, the number of feature vectors falling in each hyper-box is counted and this count is associated with the respective hyper-box creating the required histogram.

After the histogram is generated in the feature space, a peak-climbing clustering approach is utilized to group the features into distinct clusters. This is done by locating the peaks of the histogram. In Figure 1, this peak climbing approach is illustrated for a two-dimensional space example.

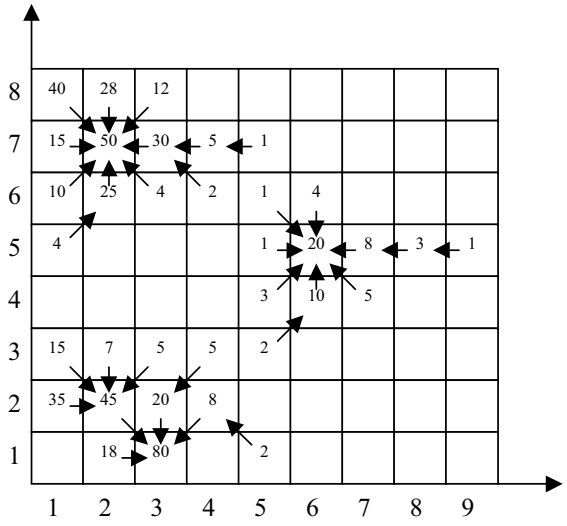

Figure 1. Illustration of the peak climbing approach for a two-dimensional feature space example

The number in each cell (hyper-box) represents a hypothetical count for the feature vectors captured by that cell. By examining the counts of the 8-neighbors of a particular cell, a link is established between that cell and the closest cell having the largest count in the neighborhood. At the end of the link assignment, each cell is linked to one parent cell, but can be parent of more than one cell. A peak is defined as being a cell with the largest density in the neighborhood, i.e. a cell with no parent. A peak and all the cells that are linked to it are taken as a distinct cluster representing a mode in the histogram. Once the clusters are found, these can be mapped back to the original data domain from where the features where extracted. Features grouped in the same cluster are tagged as belonging to the same category.

4. High performance data clustering

In the clustering algorithm described in [21], the input data is an address to a data tensor in memory with dimensions J_V x J_H x N containing Multispectral Simultaneous Autoregressive (MSAR) Random Field model features from a video frame, and the number of quantization levels to be used, which applies to all dimensions of the feature space. J_V denotes the number of windows used to extract the MSAR model features in the vertical direction of the video frame, and J_H is the number of windows in the horizontal. For convenience let us denote the total number of feature vector as $J = J_V$ x J_H. The input is fixed point data normalized in the interval [-1, +1) over the entire J_V x J_H x N data set. The output of the architecture is an address to a J_V x J_H matrix containing the clusters in the video frame space and the number of resulting clusters.

Figure 2 shows the different steps of this implementation of the clustering algorithm and the overall architecture. The chosen architecture follows a globally systolic partition. These steps are the computation of the mimum and maximum values of each dimension of the feature vectors, finding the cell size CS(k) for each k dimension, creating the histogram or assigning bin indexes to the data vectors, allocating the vectors to bin numbers, link the bins, assign the clusters, group the clusters in parallel of determining the number of clusters, and mapping the clusters back to the video frame spatial domain from the multidimensional MSAR feature space.

In order to achieve an ultra high speed implementation that supports real-time high density video, there are a number of design considerations as follows:

1. Algorithm implementation analysis and benchmarking in C to group the major step into phases that can borrow budgeted time from each other.
2. Parallelization of steps whenever possible as in the computation of the maximum and minimum values over the data set in each dimension.
3. Using register bank architectures that maximize parallel data access.
4. Dual access to pipeline register banks and a large global memory structure for redundant access storage of data through the different stages of the computation.

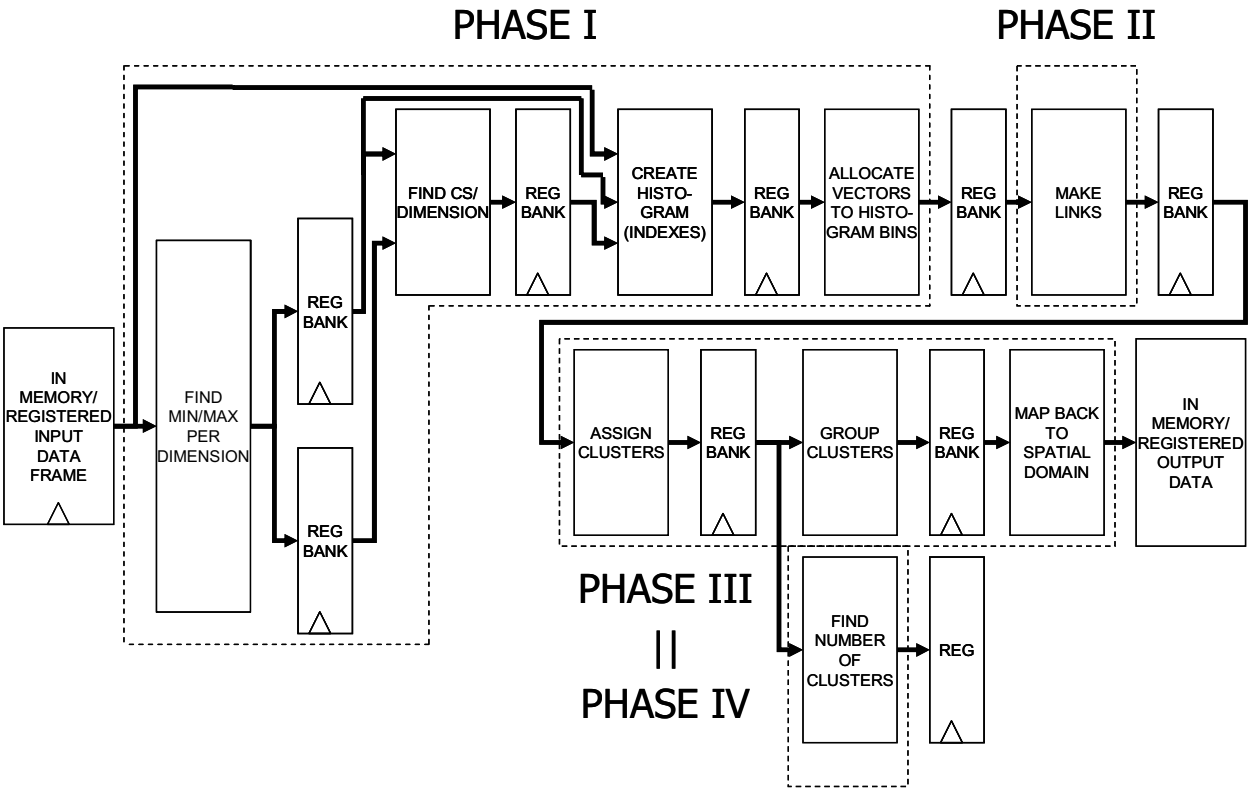

Figure 2. Peak climbing clustering algorithm overall architecture

5. Architectural details

This section presents the architectural details of this high speed data clustering processor. In all figures, the Processing Element (PE) being discussed is bounded by dashed lines. Figure 3 shows the PE for the operation of finding the minimum and maximum values for each dimension of the feature vectors in the data set. N PEs are

instantiated in parallel; one for each dimension. The operations to find the minimum and maximum values are run sequentially, thus making use of a single MIN/MAX cell in the PE. Each instantiated PE cycles J times through all the values in a given dimension of the feature vectors.

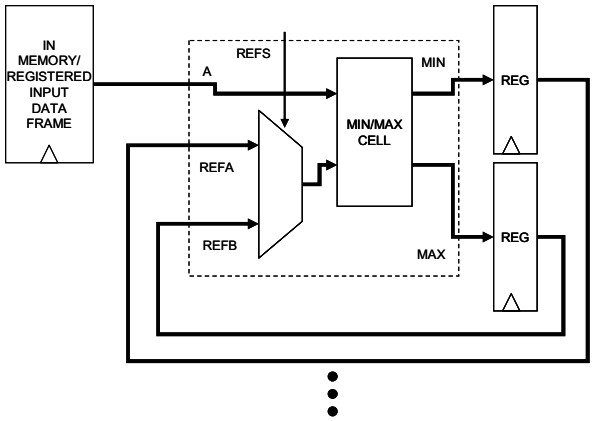

Figure 3. Min-Max processing element

Figure 4 shows the details of the PE to compute the Cell Size CS(k) for each dimension. N PEs are instantiated in parallel; one for each dimension. Because of the high dimensionality of Random Field models, the number of quantization levels in each dimension necessary for effective and efficient clustering is very small; Q = 3 … 8. This allows the division operation of Equation 1 to be implemented by a multiplication by the inverse of Q stored in a small look-up table (LUT).

Figure 5 shows the details of the PE to compute histogram indexes for each data vector. N PEs are instantiated in parallel; one for each dimension, and each instantiated PE cycles J times through all the values in a given dimension of the feature vectors. Since the possible number of quantization levels has been constrained to six, the division in the Index PE can be implemented by simple parallel restoring division algorithm that has been limited to computing only the first three bits of the quotient.

Figure 6 shows the details of the PE to allocate and identify a data vector with a given histogram bin. J instantiations of this PE are made, which corresponds to one instantiation per each possible bin. The purpose of the compressor is to count the number of ones from the comparators, which corresponds to the density of a given bin in the histogram. Table 1 additionally shows the specific structure of the compressor tree with full adders and half adders capped at J for J = 24882.

6. Discussion and conclusions

The rest of the micro-architecture to establish the links between the histogram bins, and assign the clusters, so that the results can be output, follow a very similar structure as Figure 6. The only notable exception is that the PE uses a novel computational cell to calculate the norm between two 22-dimensional vectors. This cell is shown in Figure 7. A global clocking, control, and shared memory network tie all the modules together to form the complete architecture.

This paper describes a high performance VLSI architecture for the real-time clustering of high dimensionality data extracted from video. Processing rates suitable for DVD quality video processing at MPEG-2 frame rates can be sustained.

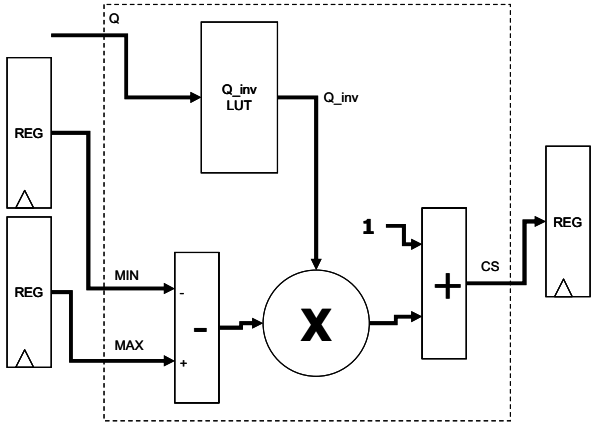

Figure 4. Cell size CS(k) processing element

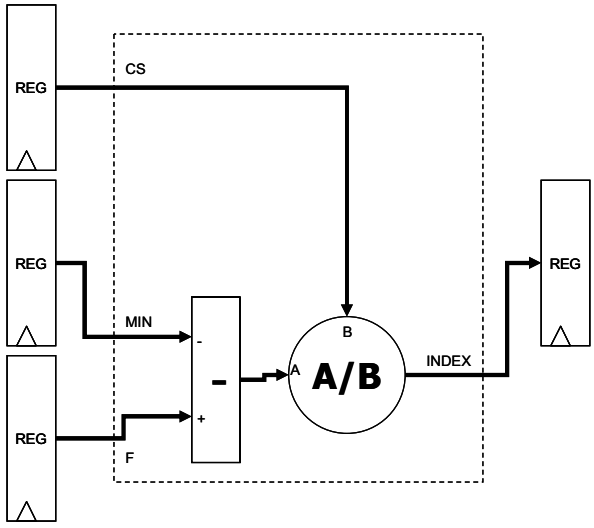

Figure 5. Index processing element

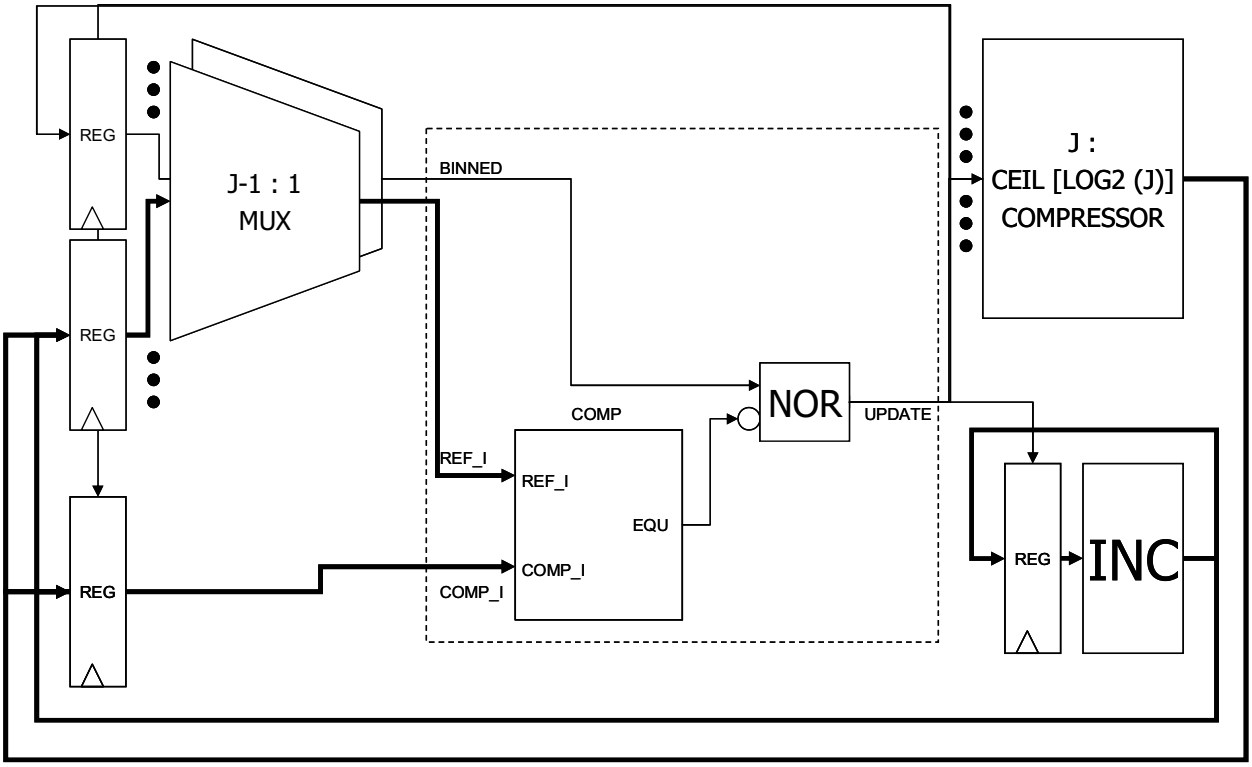

Figure 6. Processing Element used to allocate vectors to histogram bins

Table 1. Compressor structure

MODULE DESC.	MODULE NAME	OPERATION ON PREVIOUS LAYER/STAGE	NUMBER OF MODULES	INPUTS	OUTPUTS	MAXIMUM OUTPUT VALUE
INPUTS	INPUTS	NONE	24882	24882b	1b	1
FA	fa	/3	8294	3b	2b	3
HA	ha_2b	/2	4147	2b	3b	6
HA	ha_3b	/2, round down	2073	3b	4b	12
HA	ha_4b	/2, round up	1037	4b	5b	24
HA	ha_5b	/2, round down	518	5b	6b	48
HA	ha_6b	/2	259	6b	7b	96
HA	ha_7b	/2, round up	130	7b	8b	192
HA	ha_8b	/2	65	8b	9b	384
HA	ha_9b	/2, round down	32	9b	10b	768
HA	ha_10b	/2	16	10b	11b	1536
HA	ha_11b	/2	8	11b	12b	3072
HA	ha_12b	/2	4	12b	13b	6144
HA	ha_13b	/2	2	13b	14b	12288
HA	ha_14b	/2	1	14b	15b	24576
HA	ha_15b	/2, round up	1	15b	16b	48705*

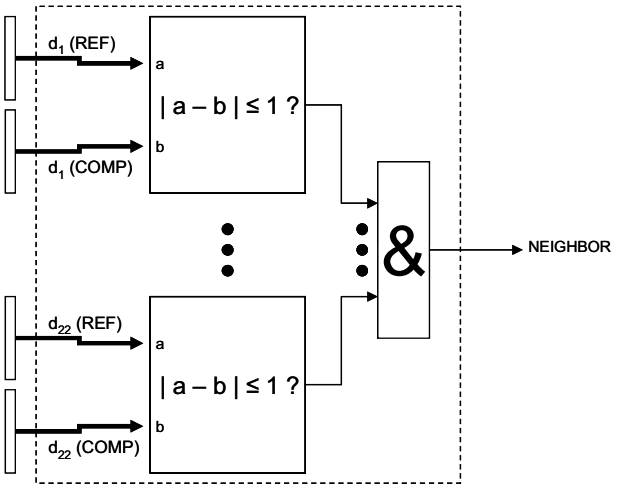

Figure 7. Neighbor detector

By using a top level quasi-systolic architectural partitioning scheme and extensive connectivity at the lower levels, the performance is improved 118 times or more than what can be achieved in a generic compute platform. This architecture can be used in many military, industrial, and commercial applications that require real-time intelligent machine processing of high quality video.

7. References

[1] A. Khotanzad and O. J. Hernandez, "Color Image Retrieval Using Multispectral Random Field Texture Model & Color Content Features," *Pattern Recognition Journal*, vol. 36, no. 8, August 2003, pp. 1679-1694.

[2] O. J. Hernandez, "Color Image Retrieval Using Multispectral Random Field Texture Model and Color Content Features," Ph.D. Dissertation, *Southern Methodist University*, May 2002.

[3] J. W. Bennett and A. Khotanzad, "Multispectral Random Field Models for Synthesis and Analysis of Color Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, March 1998, pp. 327-332.

[4] J. W. Bennett, "Modeling and Analysis of Gray Tone, Color, and Multispectral Texture Images by Random Field Models and Their Generalizations," Ph.D. Dissertation, *Southern Methodist University*, May 1997.

[5] Y. Miyanaga, M. Teraoka, and K. Tochinai, "Parallel and Adaptive Clustering Method Suitable for a VLSI System," *Proceedings – IEEE International Symposium on Circuits and Systems*, vol. 1, no. 1, June 11-14 1991, pp. 356-359.

[6] M.-F. Lai, C.-H. Hsieh, and Y.-P. Wu, "A VLSI Architecture for Clustering Analyzer Using Systolic Arrays," *Proceedings of the 12th IASTED International Conference on Applied Informatics*, May 18-20 1994, p. 260.

[7] M.-F. Lai, Y.-P. Wu, C.-H. Hsieh, "Design of Clustering Analyzer Based on Systolic Array Architecture," *Proceedings of the 1994 IEEE Asia-Pacific Conference on Circuits and Systems*, December 5-8 1994, p. 67-72.

[8] M.-F. Lai, M. Nakano, Y.-P. Wu, and C.-H. Hsieh, "VLSI Design of Clustering Analyzer Using Systolic Arrays," *IEE Proceedings: Computers and Digital Techniques*, vol. 142, no. 3, May 1995, pp. 185-192.

[9] M.-F. Lai and C.-H. Hsieh, "A Novel VLSI Architecture for Clustering Analysis, "*Proceedings of the 1996 IEEE Asia Pacific Conference on Circuits and Systems*, November 18-21 1996, pp. 484-487.

[10] R. Doallo and E. L. Zapata, "A VLSI Systolic Architecture for Solving DBT-Transformed Fuzzy Clustering Problems of Arbitrary Size," *Parallel Computing*, vol. 13, no. 3, March 1990, pp. 321-335.

[11] T. Serrano-Gotarredona and B. Linarea-Barranco, "A Real-Time Clustering Microchip Neural Engine," *IEEE Trans. on Very Large Integration (VLSI) Systems*, vol. 4, no. 2, June 1996, pp. 195-209.

[12] J. Sitte, T. Körner, and U. Rückert, "An Analog-Current Mode Local Cluster Neural Net," *Proceedings of the 1997 IEEE 6th International Conference on Emerging Technologies and Factory Automation*, September 9-12 1997, pp. 237-242.

[13] F. Perez and C. Koch, "Toward Color Image Segmentation in Analog VLSI: Algorithm and Hardware," *International Journal of Computer Vision*, vol. 12, no. 1, February 1994, pp. 17-42.

[14] D. D. Zhang, "System Design Methodology for Fuzzy Clustering Neural Networks," *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, October 14-17 1996, pp. 1062-1066.

[15] D. Zhang, M. Kamel, and M. I. Elmasry, "Fuzzy Clustering Neural Network System Design and Implementation," *Midwest Symposium on Circuits and Systems*, vol. 2, no 2, August 3-5 1994, pp. 1381-1384.

[16] D. Zhang and S. K. Pal, "A Fuzzy Clustering Neural Networks (FCNs) System Design Methodology," *IEEE Trans. on Neural Networks*, vol. 11, no 5, September 2000, pp. 1174-1177.

[17] P. Poiré, Y. Savaria, H. Daniel, M.-A. Cantin, and Y. Blaquière, "Hardware/Software Codesign of a Fuzzy ART Neural Clusterer: The Benefits of Configurable Computing," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 3526, November 1998, pp. 90-96.

[18] A. Granger, Y. Blaquière, Y. Savaria, M.-A. Cantin, and P. Lavoie, "A VLSI Architecture for Fast Clustering with Fuzzy ART Neural Networks," *Proceedings of International Workshop on Neural Networks for Identification, Control, Robotics, and Signal/Image Processing, NICROSP*, August 1996, pp. 117-125.

[19] M.-A. Cantin, Y. Blaquière, Y. Savaria, A. Granger, and P. Lavoie, "Implementation of the Fuzzy ART Neural Network for Fast Clustering of Radar Pulses," *Proceedings - IEEE International Symposium on Circuits and Systems*, vol. 2, May 1998, pp. 458-461.

[20] J. Sitte, T. Körner, and U. Rückert, "Local Cluster Neural Net Analog VLSI Design," *Neurocomputing*, no. 19, 1998, pp. 185-197.

[21] A. Khotanzad and A. Bouarfa, "Image Segmentation by a Parallel, Non-Parametric Histogram Based Clustering Algorithm," *Pattern Recognition Journal*, vol. 23, no. 9, September 1990, pp. 961-963.