Design of a Classification System for Rectangular Shapes Using a Co-Design Environment

Rolf F. Molz[1], Paulo M. Engel[1], Fernando G. Moraes[2], Lionel Torres[3], Michel Robert[3]

[1]*UFRGS – Instituto de Informática*
Av. Bento Gonçalves, 9500 – Bloco IV
Caixa Postal 15064
91501-971 - Porto Alegre – Brazil
{rolf,engel}@inf.ufrgs.br

[2]*PUCRS – Faculdade de Informática*
Av. Ipiranga, 6681 - Prédio 30
90619-900 - Porto Alegre - Brazil
moraes@inf.pucrs.br

[3]*LIRMM –Université Montpellier II*
161, rue Ada
34392 Montpellier - Cedex 5 - France
{torres,robert}@lirmm.fr

Abstract

Pattern localization and classification are CPU time intensive, being normally implemented in software. Custom implementations in hardware allow real-time processing. In practice, in ASIC or FPGA implementations, the digitization process introduces errors that should be taken into account. This paper presents initially the state-of-the-art in this field, analyzing the performance and implementation of each work. After we propose a system for rectangular shapes localization and classification using reconfigurable devices (FPGA) and a signal processor (DSP) available in a flexible codesign platform. The system will be described using C and VHDL languages, for the software and hardware parts respectively. Finally, it is described the classification block of the system, implemented as an Artificial Neural Network (ANN) in a rapid prototyping platform.

1. Introduction

The recognition of an object in an image is a complex task that involves a broad range of techniques. Various steps involved in an image-understanding system are shown in Figure 1. The system consists of six stages: image acquisition, preprocessing, feature extraction, association storage, knowledge base, and recognition/classification. These stages essentially correspond to the low-level, intermediate-level, and high-level processing.

This paper proposes an image processing system, aiming the integration of a Digital Signal Processor (DSP) with programmable devices (FPGA). The system comprises preprocessing, feature extraction and classification tasks. These tasks will be employed to find and classify rectangular shapes for an input image. These rectangular shapes can be traffic signals, landmarks for robotics, car license, and so on.

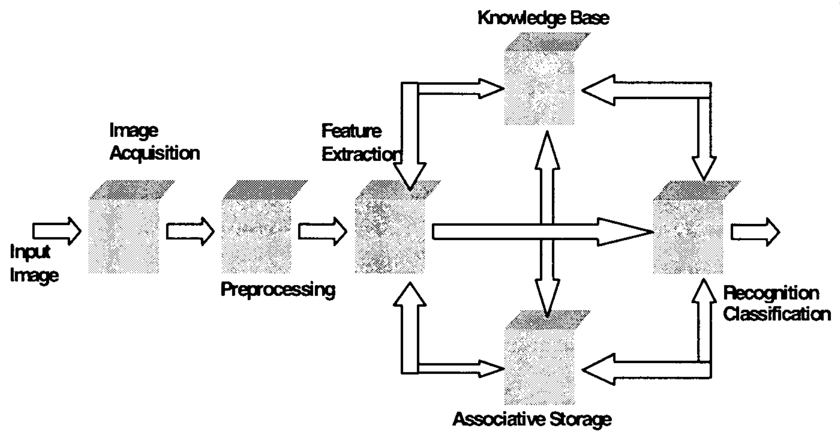


Figure 1. Model for image recognition system

The DSP processor is used when sequential processing is required or floating point arithmetic is needed. The programmable device is used for parallel and bit level operations. This partition improve the performance of the system when compared to pure DSP implementations, reduce the time to market and cost when compared to ASIC implementation. So, this hardware/software system represents a good trade-off between performance and cost.

This paper is organized as follows. Section 2 presents the state-of-the-art in object localization and classification, describing the function, methodology, implementation (software, mixed, ASIC or FPGA) and performance of each system. Next, in Section 3, we describe the system structure, detailing the tasks that must be accomplished. Section 4 presents preliminary results of the hardware implementation, concerning the classification task. Finally we present our conclusion and future work.

2. State-of-the-art

In this section, we describe the state-of-the-art in systems dealing with object localization and classification. Table 1 presents relevant works published in the literature. After this table, a comparative analysis between these systems and our work is made.

Table 1. State-of-the-art summary

Ref.	Function	Methodology	Implementation	Performance
[1]	Detection and recognition of transit signal or pedestrian	Distance Transforms and hierarchical tree are used to detection. Recognition by RBF	Dual-Pentium II 450MHz MMX	10-15 frames / sec (360 x 288 pixels). Pedestrian: 1 -5 frames / sec.
[2]	Objects detection and recognition with occlusion	Several stages of the pre-processing. Classification by MLP	Software implementation	3 minutes / image
[3]	Objects recognition with occlusion	Dominants points Detection. Recognition by Euclidean distance	Software implementation	
[4]	Objects detection and recognition	Detection by background extraction and squeletization Recognition by data base.	Software implementation Pentium II - 300MHz	3 frames / sec
[5]	Objects detection and recognition with occlusion	Gaussian filter for pre-processing. Dominants points extraction. Recognition by SIFTs combination.	Software implementation Sun Sparc 10	1 image (384 x 512 pixels) in 2 sec
[6]	Objects localization	Hausdorff Distance	Software implementation 8 processadores Sun SPRAC Server 1000	Example box = +- 7min.
[7]	Danger signals detection and classification	Detection and classification by pre-processing and an RAM type ANN	Software implementation Pentium 486-66MHz	10 sec (detection and classification)
[8]	License vehicle detection and recognition	Pre-processing to detection and recognition by MLP	Pre-processing by FLEX10K100 Recognition by PC	30 frames / sec
[9]	Landmarks detection and recognition	Recognition by MLP ANN	Comparison: PC, Sparc20, PAPRICA.	400ms / image (2,5 frames / sec.)
[10]	Neural architecture only		ASIC - 1,2μm double-poly CMOS analog	12x10^6 sinapses / sec.
[11]	Generic images processing		4 chips PAPRICA-3 ASIC 0,8μm CMOS	
[12]	Time-real image processing by ASICs		Many ASICs	
[13]	Lines extraction by Hough Transform		Board with FPGAs	
[14]	Image squeletization	Comparison between hardware and software		
[15]	Dominants points and object outline		ASIC (0,8μm) and FPGA	
[16]	Time-real segmentation		ASIC 0,8μm double-metal	30 frames/sec
[17]	FPGA convolution		FPGA (2 x XC4013)	14 x more than DSP C-40
[18]	Pattern classification		ASIC 1,2μm and FPGA	ASIC < 100ns / point
[19]	Edge detection		ASIC 1,0μm	30ns / pixel

Where: ref.: reference

Among these works, the closest to our goal are [1] to [9]. The works [1], [7], [8] and [9] have the great affinity to our proposed system.

In [9] the objects must be in pre-defined positions and is completely implemented in software. A hardware implementation using the PAPRICA ASIC connected to a host is also presented in [9]. The reported results do not allow real time processing (2.5 frames/sec). The work [8] assumes a fixed distance between the vehicle and the image sensor. This restriction allows the use a proportional rule in order to accelerate the detection of desired patterns in the vehicle license. This system has a mixed implementation, with the software part running in a PC, and the hardware part running in a FPGA. The system can be used in real-time applications (30 frames/sec). The work [7] uses ANN for classification, however the system is implemented in software, resulting in a poor performance (10 sec for localization and classification). Finally, a similar work is presented in [1], aiming object localization and classification, however it was implemented in software (10-15 frames/sec). These works do not include object occlusion.

Our objective is to develop a portable system for image processing, having light weight and low power consumption. The performance for real time processing is obtained implementing the parallel operations in hardware, using FPGAs. This hardware/software implementation is a full stand-alone system, able to execute all required tasks for rectangle shape localization and classification. This system can implemented in a dedicated ASIC, characterizing a system-on-a-chip for image processing.

To complete our system, we can connect it to a CIS circuit (CMOS Image Sensor) in order to include the image acquisition task. Finally, the learning phase can be implemented in software, increasing the flexibility of the system.

3. Proposed system

In this section, the tasks for rectangle shape localization and classification are described. The ANN executes the classification using learned shapes.

We can find in the literature some methods for geometrical shapes localization (objects): texture analysis [20], fuzzy logic [21], local features [22], global features of the desired shapes (polygons approximation [23] and dominants points detection [24]) and mathematical morphology [25]. In this work we used the *global features of the desired shapes* method. As the objects we want to localize and classify have rectangular shape, this simple method can be used. The system to implement such method, *global features of the desired shapes*, is depicted in Figure 2, as a sequence of independent tasks.

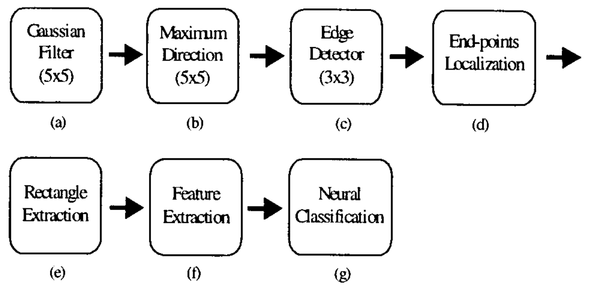


Figure 2. Block diagram of the proposed system

The first task to obtain the *global features of the desired shapes* is the segmentation or line extraction process. In [26] several methods for line extraction in images are presented, and a comparison between their performance is done. The works [26] and [27] relate that the Canny method presents the best results among several algorithms, however its CPU time consuming is too elevate to consider real-time processing. An alternative approach, [28], presents a method suited for hardware implementation. This algorithm performs line extraction and segmentation using the gradient and direction of each pixel into the source image.

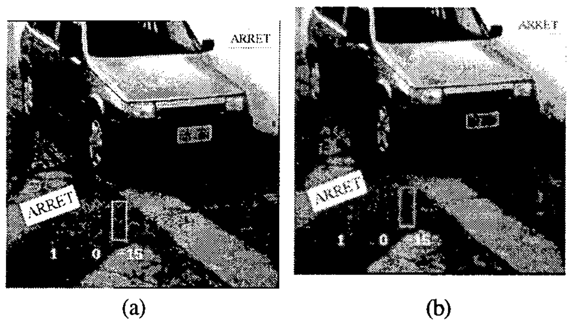
The segmentation process starts applying a Gaussian Filter over the input image (Figure 2a). This filter has the smoothing function in the input image. In this work we used a 5x5 window. Equation 1 describes this filter. A fixed-point constraint is applied to this filter, since it will be implemented in hardware. This filter can be seen as smoothing operation.

$$g'(i,j)=\sum_{p=-m}^{m}\sum_{q=-n}^{n}w(p,q)*g(i-p,j-q) \qquad (1)$$

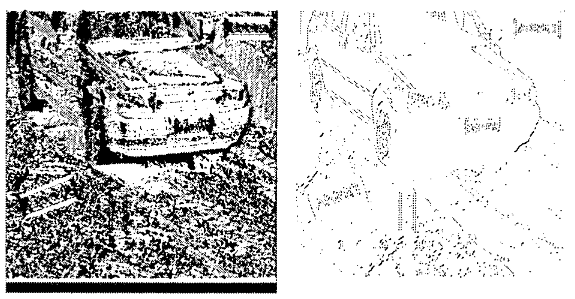
Where: w (p, q) = weighting coefficients;
g (i-p, j-q) = gray value of the (i-p , j-q) pixel;
g' (i, j) = new gray value of the (i, j) pixel;
m, n = width and height window.

Next, each pixel direction is obtained, using a 5x5 window (Figure 2b). Each pixel can have one of four different directions: horizontal, vertical, positive and negative slope. The third step applies a threshold value over the image containing the pixel direction, using a 3x3 window (Figure 2c). The resulting image contains the edges of the original image, with its directions.

Figure 3 illustrates these three steps. Figure 3a presents the original image, Figure 3b the image after the Gaussian filter, Figure 3c the image with the pixel directions and in the Figure 3d the image with the edges obtained from Figure 3c.



(a) (b)



(c) (d)

Figure 3. Segmentation process

The segmentation process (steps a, b and c in Figure 2) does not localize the lines. The algorithm detects the existence of lines, with some dimension and some angle inside the image. To obtain the exact localization of these lines two approaches can be considered:

1) **Vertices localization**. Pre-defined convolution masks are applied in the image to obtain the vertices localization. This process calculates the straight angles in this image. The convolution masks works only for straight angles or vertices with zero near values. This method can not be used, since the image acquisition process can introduce some noise, making difficult to find straight angles.. Others techniques for this method can be used, such as [29], [30] and [31].

2) **Lines end-points localization**. This approach is used by [32]. This method can be easily implemented, and has a good performance. However, a good segmentation process is necessary for the end-points localization algorithm. This algorithm is the task (d) in Figure 2. The output of this algorithm gives the initial position, final position, angle and length for each line in the input image. Figure 4 illustrates the image obtained by this algorithm.

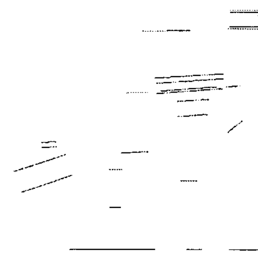


Figure 4. Lines obtained using the end-point localization algorithm

After, Figure 2e, the rectangles are extracted associating lines with similar angles, length and initial position (in X axe direction). The result is shown by Figure 5.

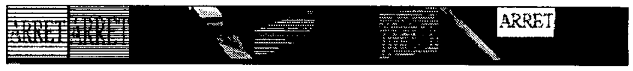


Figure 5. Image with the rectangle shapes

The sixth task of our system is the "feature extraction" (Figure 2f). This extraction comprises angle correction and viewport mapping (50 x 50pixels) [33]. Next, the averages in the viewport columns are calculated. These average values are applied in the classification process, Figure 2g.

A MLP (Multi-Layer Perceptron) Artificial Neural Network (ANN) with supervised learning accomplishes the classification process. The method to correct errors is the backpropagation algorithm.

4. Implementation Analysis and Preliminary Results

The block diagram presented in Figure 6 corresponds to the hardware implementation of our system.

The system is organized as a pipeline, to increase the final throughput of the system. The tasks executed in hardware, by the FPGA device are Gaussian filter, maximum direction computation, edge detection, and end-point localization. Between each pipeline stage (task), buffers are inserted. Between the Gaussian filter and direction computation a buffer with 5 image lines is required. So, the direction computation starts after the Gaussian Filter has processed 5 lines. Between the edge detection and direction computation a buffer with 3 image lines is inserted, corresponding to edge detection window. The edge detection writes its results in the FPGA external memory. The end-point localization reads the memory, sending each processed line to the DSP.

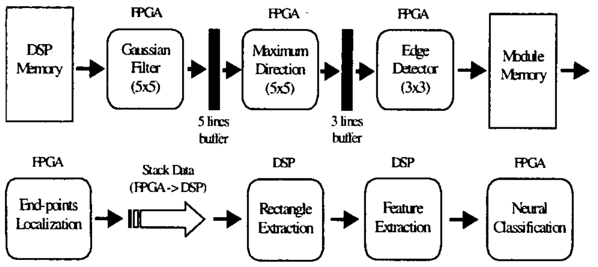


Figure 6. System blocks diagram

The DSP processes the tasks corresponding to rectangle extraction and feature extraction. We consider in the future implement the rectangle extraction in hardware. The feature extraction is done in the DSP, since a huge amount of mathematical operations are executed. The work [34] present some methods for feature extraction. These feature values are sended to classification block implemented in hardware.

The ANN is used for the classification block. This ANN is implemented in hardware (FPGA) assuring a great output performance due the parallel operations [35]. This block was implemented in a codesign platform. The APTIX [36] board is used for this implementation. This board has two FPGA modules (10k100 - Altera) and one DSP core module (D950 - ST Microelectronics).

Figure 7 presents the blocks diagram of the neuron implemented. This neuron is controlled by the DSP, which sends inputs and weights. This control will be implemented in hardware, since a simple state machine can carry out this task.

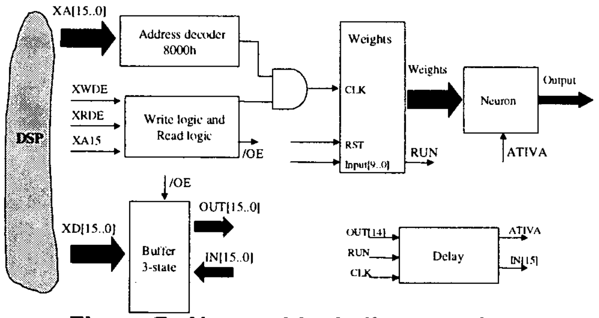


Figure 7. Neuron block diagram of the classification process

The classification module uses 1453 LE (Logic Elements), consuming 29% of the 10K100 FPGA. This large hardware consumption is due to the parallel processing, i.e., the neuron processes 11 input patterns in parallel. After neuron propagation, a sigmoidal activation function is applied. The activation function is implemented using LUTs (Look up Table). The complete description of the neuron implementation (propagation and output function) can be found in [35].

5. Conclusions

This paper discussed the implementation of a complete system for rectangle shapes localization and classification. We presented preliminary results concerning the classification block, implemented in programmable devices (FPGAs) of a codesign platform.

This first implementation of the classification block has a maximum frequency of 5,8MHz in the Altera device. Despite of this low frequency, the system can be faster than DSP processors, since parallel operations are executed in this hardware block. The others system blocks are currently being implemented.

All presented algorithms to solve this problem are fast and suitable to hardware implementation, presenting small error when compared to software implementations (for example, the software implementations do not have fixed-point restrictions). However, occlusion shapes are not considered by our system. This feature need others processing methods, such as [22], [32], [23] and [24].

The proposed work can be a future prototype platform for image processing, allowing implementing and testing several algorithms. Also, each developed algorithm in VHDL can be inserted in library, composing in this way a rich environment to implement customized image processing systems.

Acknowledgements

The author Rolf Fredi Molz would like to gratefully acknowledge the support of CAPES Capes/Cofecub project number 245/98-Brazil. The author Fernando G. Moraes would like to gratefully acknowledge the support of CNPq project number 522939/96-1-Brazil.

References

[1] GAVRILA, D.M. ; PHILOMIN, V. Real-Time Object Detection for "Smart" Vehicules. **International Conference on Computer Vision (ICCV99)**. Vol. 1. Corfu, Greece, 20-25 september, 1999.

[2] SIM, H.C.; NG. G.S. Recognition of Partially Ocluded Objects with Back-Propagation Neural Network. **Intern. Journal of Pattern Recognition and Artificial Intelligence**, vol. 12, n°. 5, pp. 645-660, 1998.

[3] TSANG, K.M.; TO, F.W. Recognition of Partially Occluded Objects using an Orthogonal Complex AR Model Approach. **Intern. Journal of Pattern Recognition and Artificial Intelligence**, vol. 13, n°. 1, pp. 85-107, 1999.

[4] FORESTI, G.L. Object Recognition and Tracking for Remote Video Surveillance. **IEEE Trans. On Circuits and Systems for Video Technology**, vol. 9, n° 7, pp. 1045-1062, october 1999.

[5] LOWE, David G. Object Recognition from Local Scale-

Invariant Features. **Proc. Of the International Conference on Conputer Vision**, Corfu, Greece, 20 - 25 September, 1999.

[6] RUCKLIDGE, William. Efficiently Locating Objects using the Hausdorff Dsitance. **International Journal of Computer Vision** vol. 24, n° 3, pp. 251-270, 1997.

[7] JORGENSEN, T.M.; CHRISTENSEN, S.S.; ANDERSEN, A.W. Detecting danger labels with RAM-based neural networks. **Pattern Recognition Letters**, vol. 17, pp. 399-412, 1996.

[8] MINGO, Ferran L. **Configurable Computing for Real-Time Vision**. Tése de Doutorado, Universitat Autônoma de Barcelona - UAB, pp. 180, Bellaterra, september, 1998.

[9] ADORNI, G.; GORI, M.; MORDONINI, M. Just-in-time Landmarks Recognition. **Real-Time Imaging**, vol. 5, pp. 95-107, 1999.

[10] HAN, Gunhee; SÁNCHEZ-SINENCIO, Edgar. A Flexible and Expendable Neuroimage Processor Architecture. **IEEE Transactions on Circuits and Systems - I: Fundamental Theory and Applications**, vol. 46, n° 9, pp. 1055-1063, september, 1999.

[11] SANSOÉ, C.; GREGORETTI, F.; REYNERI, L.M. A Neuro-Fuzzy Real-Time Image Processing System. **Proceedings of the 25th Euromicro Conference**, Milan, Italy, 8 - 10 September, 1999.

[12] KRALJIC, I.C.; QUENOT, G.M.; ZAVIDOVIQUE, B. Investigating real-time validation of Real-Time Image Processing ASICs. **Proceedings of the Computer Architectures for Machine Perception (CAMP-97)**, Como, Italy, October 20-22, 1997.

[13] MAHMOUD, M.; NAKANISHI, M.; OGURA, T. Hough transform implementation on a reconfigurable highly parallel architecture. **Proceedings of the 1997 Computer Architectures for Machine Perception (CAMP)**, 1997.

[14] SUDHA, N.; NANDI, S. A Parallel Skeletonization Algorithm and its VLSI Architecture. **Proceedings of the Fifth Internatinal Conference on High Performance Computing**, Madras, India, 17-20 December, 1998.

[15] DALLAIRE, S.; TREMBLAY, M.; POUSSART, D. VLSI Architectures for the Embedded Extraction of Dominant Points on Object Contours. **Proceedings of the Computer Architectures for Machine Perception (CAMP-97)**, Como, Italy, October 20-22, 1997.

[16] ALZAHRANI, F.M.; CHEN, T. A Real-Time Edge Detector: Algorithm and VLSI Architecture, **Real-Time Imaging**, vol. 3, pp. 363-378, 1997.

[17] BOSI, B.; BOIS, G.; SAVARIA, Y. Reconfigurable Pipelined 2-D Convolvers for Fast Digital Signal Processing. **IEEE Trans. On Very Large Scale Integration (VLSI) Systems**, vol. 7, n°. 3, Sept. 1999.

[18] ROBERT, M; GORRIA, P.; MITÉRAN, J.; TURGIS, S. Architectures for a Real Time Classification Processor. **Conference on Custom Integrated Circuits**, 1994.

[19] TORRES, L.; BOURENNANE, E.; ROBERT, M.; PAINDAVOINE, M. A Recursive Digital Filter Implementation for Noisy and Blurred Images. **Real Time Imaging Journal**, vol. 4, n° 3, pp. 181-191, June, 1998.

[20] LIBERMANN, F. Classificação de Imagens Digitais por Textura usando Redes Neurais. **Dissertação de Mestrado** – Instituto de Informática – UFRGS/ Brasil, 86 pp. 1997.

[21] HOEPPNER, F. Fuzzy Shell Clustring Algorithms in Image Processing: Fuzzy C-Rectangular and 2-Rectangular Shells. **IEEE Transactions on Fuzzy Systems**, vol. 5, n. 4, pp. 599- 613, November 1997.

[22] LOWE, David G. Object Recognition from Local Scale-Invariant Features. **Proc. Of the International Conference on Conputer Vision**. Sept. 1999.

[23] YANG, Fan. Traitement automatique d'images de visages algorithmes et architecture. **Thèse de Doctorat** – U.F.R. Sciences et Techniques -Université de Bourgogne /France, 181pp., Juin 1998.

[24] TSANG, K.M.; TO, F.W. Recognition of Partially Occluded Objects using an Orthogonal Complex AR Model Approach. **Intern. Journal of Pattern Recognition and Artificial Intelligence**, vol. 13, n°. 1, pp. 85-107, 1999.

[25] FORESTI, G.L. Object Recognition and Tracking for Remote Video Surveillance. **IEEE Trans. On Circuits and Systems for Video Technology**, vol. 9, n° 7, oct. 1999.

[26] HEATH, M.; SARKAR, S.; SANOCKI, T.; BOWYER, K. Comparison of Edge Detectors. **Computer Vision and Image Understanding**, vol. 69, n. 1, pp. 38-54, January, 1998.

[27] PARKER, J.R. **Algorithms for Image Processing and Computer Vision**. John Wiley & Sons, Inc. New York. 1997.

[28] ALZAHRANI, F.M.; CHEN, T. A Real-Time Edge Detector: Algorithm and VLSI Architecture, **Real-Time Imaging**, vol. 3, pp. 363-378, 1997.

[29] MOKHTARIAN, F.; SUOMELA, R. Robust Image Corner Detection Through Curvature Scale Space. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, vol. 20, n. 12, pp. 1376-1381, Dec. 1998.

[30] CHABAT, F.; YANG, G.Z.; HANSELL, D.M. A corner orientation detector. **Image and Vision Computing**, vol. 17, pp. 761-769, 1999.

[31] FREEMAN, H. On the encoding of arbitrary geometric configurations. **IRE Trans. Electron. Comput**, vol. 26, pp. 297-303. 1977.

[32] COCQUEREZ, J.; PHILIPP, S.; et al. **Analyse d'images: filtrage et segmentation**. Masson, Paris, 1995.

[33] FOLEY, James D. **Computer Graphics : Principles and Practice, Second Edition in C.** 1995.

[34] KULKARNI, Arun D. **Artificial Neural Networks for Image Understanding**. Van Nostrand Reinhold, USA, 1994.

[35] MOLZ, R. F.; ENGEL, P.M. ; MORAES, F.G. "Uso de um ambiente codesign para a implementação de redes neurais", **IV Congresso Brasileiro de Redes Neurais**, p. 13-18, São José dos Campos, Brasil, July 1999.

[36] APTIX Corporation. www.aptix.com