ECCTD '95 European Conference on
Circuit Theory & Design

EDGE EXTRACTION AND ENHANCEMENT WITH COORDINATE LOGIC FILTERS USING BLOCK REPRESENTED IMAGES

Basil G. Mertzios[1], Iraklis M. Spiliotis[1], Rallis C. Papademetriou[2] and Kostantinos Tsirikolias[1]

[1]*Department of Electrical and Computer Engineering, Democritus University of Thrace, 67 100 Xanthi, Hellas*
[2]*Department of Electrical and Electronic Engineering, University of Portsmouth, Portsmouth, UK*

Abstract - **A new nonlinear class of filters, the Coordinate Logic (CL) Filters and their application in edge extraction and enhancement is presented. The CL filters are execute Coordinate Logic Operations (CLOs) among the image points and may be seen as multibinary morphological-like kind of filters. The Image Block Representation (IBR), an advantageous image representation scheme is also presented. The Coordinate Logic Filters can be easily and fast implemented using block represented images.**

I. INTRODUCTION

The CL filters are nonlinear filters that execute Coordinate Logic Operations (CLOs) [1]-[4], among the pixels of the image. If the signal is not binary, then the operations are executed among their corresponding binary values. The CLOs are the *Coordinate AND* (CAND), the *Coordinate OR* (COR), the *Coordinate XOR* (CXOR) or the *Coordinate NOT* (CNOT). The CL filters are very efficient in various 1-D and 2-D processing applications, such as lowpass and highpass filtering, erosion, dilation, opening, closing, skeletonization, region filling, coding, shape smoothing as well as edge detection, shape analysis and feature extraction.

The CL filters are related to morphological filters [5]-[8] but constitute a separate new class of nonlinear filters [9]. Indeed, the morphological filters may be seen as a class of rank order filters, which by definition involve some kind of sorting, while the CL filters do not. The CL filters can execute easily and fast the four basic morphological operations (erosion, dilation, opening and closing). Therefore, the filters are expected to be suitable for all the variety of tasks that are executed by morphological filters. On the contrary, in general the morphological filters when applied to gray level images, require additional operations.

A class of nonlinear filters that is based on Boolean operators is the generalized stack (GS) filters [10]. The difference between GS filters and CL filters is that the stack filters operate on signal levels while CL filters on binary representations. The definition of GS filters is based on threshold decomposition and Boolean operators. Threshold decomposition maps the 2^n level input signal (where n is the word length), into 2^n-1 binary signals by thresholding the original signal at each of the allowable levels. The set of 2^n-1 signals is then filtered by 2^n-1 Boolean operators, which are constrained to have the stacking property. The multilevel output is finally obtained as the sum of the 2^n-1 binary output signals. On the contrary, CL filters decompose the signal in n binary signals which operate in parallel.

The most common image representation format is the two-dimensional (2-D) array. However, many research efforts for deriving alternative image representations have been motivated by the need of fast processing of huge amount of data. Such image representation approaches aim to provide machine perception of images in pieces larger than a pixel and are separated in two categories: boundary based methods and region based methods and include quadtree representations [11], chain code representations [12], contour control point models [13], autoregressive models [14] and interval coding representation [15]. Recently, a region based method, which is called image block representation has been presented [16]-[19]. In the image block representation process the whole binary image is decomposed in a set of rectangular areas with object level, which are called *blocks* and exploits the fact that many compact areas of a given binary image have the same value. In gray level images the image block representation is applied, considering n different binary images, where 2^n is the number of gray levels.

II. COORDINATE LOGIC OPERATIONS ON DIGITAL SIGNALS

The underlying idea in Coordinate Logic image processing is the execution of Coordinate Logic Operations (CLOs) among the gray level pixels. These CLOs are executed among the corresponding binary bits of equal length of the considered pixels, without counting the carry bits.

Assume two numbers A and B in decimal system, which in their binary sequence of length n are written as: $A=[a_0\ a_1\ \ldots\ a_{n-1}]$, $B=[b_0\ b_1\ \ldots\ b_{n-1}]$. The

following Definitions refer to the CLOs that are used by the CL filters.

Definition 1: Coordinate Logic Operation
The coordinate logic operation ⊙ of two numbers A and B in decimal system, in their binary sequence is given

$$C = [c_0\ c_1\ \ldots\ c_{n-1}] = A \odot B \tag{1}$$

where the i-th bit c_i is the logic operation ⊙ of the corresponding a_i and b_i bits of the operands, i.e.

$$c_i = a_i \odot b_i, \qquad i = 0,1,\ldots,n-1 \tag{2}$$

■

The operation ⊙ may be the logical OR, AND, or XOR. The distinct characteristic of the coordinate operators is that they do not encounter any carry bits as it happens in the regular logical operations among binary sequences.

Definition 2: CNOT (A)
The Coordinate Logic NOT (CNOT) operator of a number A in decimal system, in its binary sequence, is given by

$$C = [c_0\ c_1\ \ldots\ c_{n-1}] = CNOT(A) \tag{3}$$

where the i-th bit c_i is the logical NOT of the corresponding a_i bit of the number A, i.e.

$$c_i = \text{NOT}\ a_i, \qquad i = 0,1,\ldots,n-1 \tag{4}$$

■

III. DERIVATION OF THE COORDINATE LOGIC FILTERS

A 1-D digital signal may be denoted by a set in one variable, as follows:

$$S = \{s(i),\ i = 0,1,\ldots,n-1\} \tag{5}$$

while a 2-D digital signal will be denoted by a 2-D set

$$S = \{s(i,j),\ i = 0,1,\ldots,N_1-1,\ j = 0,1,\ldots,N_2-1\} \tag{6}$$

where N_1 and N_2 denote the finite dimensions of the signal in the horizontal and the vertical directions respectively.

The corresponding filtered signals are denoted by

$$F = \{f(i),\ i = 0,1,\ldots,N-1\} \tag{7}$$

and

$$F = \{f(i,j),\ i = 0,1,\ldots,N_1-1,\ j = 0,1,\ldots,N_2-1\} \tag{8}$$

respectively. At this point we give the derivation of two basic operations, the dilation and erosion using CL filters.

Definition 3: Dilation
The dilation of the image S by the structuring element B, is denoted by S_B^D and is defined by

$$S_B^D(i,j) = \text{COR}\ S(i,j) \in B = \sum_{k=0}^{n-1} (s_k)_B^D(i,j)\ 2^k \tag{9}$$

where $s_k(i,j)$, $k = 0,1,\ldots,n-1$, are the binary components of the decimal values $S(i,j)$ and $(s_k)_B^D(i,j)$ denotes the dilation operation on the binary value $s_k(i,j)$ by the structuring element B, given by

$$(s_k)_B^D(i,j) = \text{OR}\ s_k(i,j) \in B \tag{10}$$

■

Definition 4: Erosion
The erosion of the image S by the structuring element B, is denoted by S_B^E and is defined by

$$S_B^E(i,j) = \text{CAND}\ S(i,j) \in B = \sum_{k=0}^{n-1} (s_k)_B^E(i,j)\ 2^k \tag{11}$$

where $s_k(i,j)$, $k = 0,1,\ldots,n-1$, are the binary components of the decimal values $S(i,j)$ and $(s_k)_B^E(i,j)$ denotes the dilation operation on the binary value $s_k(i,j)$ by the structuring element B, given by

$$(s_k)_B^E(i,j) = \text{AND}\ s_k(i,j) \in B \tag{12}$$

■

The most fundamental properties concerning the operations of erosion, dilation, opening and closing, hold in both the binary and gray level signals and images, when they are implemented by CL filters.

IV. IMAGE BLOCK REPRESENTATION

A bilevel digital image is represented by a binary 2-D array. Without loss of generality, we suppose that the object pixels are assigned to level 1 and the background pixels to level 0. Due to this kind of representation, there are rectangular areas of object value 1, in each image. These rectangular areas have their edges parallel to the image axes and contain integer number of image pixels. At the extreme case, one pixel is the minimum rectangular area of the image. These rectangulars are called *blocks*. Fig. 1 shows an image of the character d and the blocks that constitute the image.

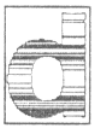

Fig. 1 Image of the character d and the blocks.

Consider a set that contains as members all the nonoverlapping blocks of a specific binary image, in such a way that no other block can be extracted from the image (or equivalently each pixel with object level belongs to only one block). This set represents the image without loss of information. In images, with 2^n gray levels the n different binary images that constitute the gray image, are represented with blocks. This

representation of the image is called *image block representation* [16]-[19].

V. LOGIC OPERATIONS ON BLOCK REPRESENTED IMAGES

When logic operations are performed in block represented binary images, the results are always block represented images. A new logic function, the COMAND function (COMplementary AND) has also been defined, which is used as an auxiliary operator for the execution of the OR, XOR and NOT operations in block represented images [19].

Definition 5: COMAND logic function

Consider the ordered pair of binary or logic operands (a,b). The COMAND logic function is defined by the formula:

$$\text{COMAND}(a,b) = a \text{ XOR } (a \text{ AND } b) \quad (13)$$

■

Definition 6: COMAND operation among binary arrays

Consider the 1-D arrays of binary operands $A(i) = [a_0, a_1, ..., a_{n-1}]$ and $B(i) = [b_0, b_1, ..., b_{n-1}]$. The COMAND operation is defined by:

$$C(i) = \text{COMAND}(A(i), B(i)) = [c_0, c_1, ..., c_{n-1}] \quad (14)$$

where

$$c_i = \text{COMAND}(a_i, b_i) = a_i \text{ XOR } (a_i \text{ AND } b_i) \quad (15)$$

■

The COMAND$(A(i),B(i))$ operation, among the 1-D arrays $A(i)$, $B(i)$ produces an equidimensional array $C(i)$, whose logical one elements are those of $A(i)$ that do not belong to $B(i)$.

The COMAND operation among multidimensional arrays is similarly defined. According to the Definition 5 and 6, it is clear that the commutative property in the COMAND operation is not preserved. The execution of the COMAND function is easily implemented with blocks that represent 2-D arrays, as it is illustrated in Fig. 2.

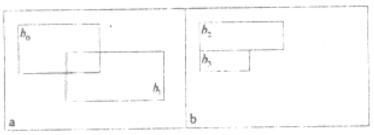

Fig. 2 (a). The relative positions of the blocks b_0, b_1 according to a rectangular grid and (b). The execution of the function COMAND(b_0,b_1), produces the blocks b_2, b_3

In the sequel we describe the implementation of the OR, AND, XOR and NOT operations among the blocks in block represented images.

The OR operation between the blocks b_0 and b_1 is executed according to the following formula:

$$b_0 \text{ OR } b_1 = \text{COMAND}(b_0, b_1) \bigcup b_1 \quad (16)$$

where the symbol of the set union $\bigcup$ means that the block b_1 is contained in the results.

Thus, the OR operation between the block represented images f_1 and f_2, is the extension of the OR operation between two blocks and is implemented according to the formula:

$$f_1 \text{ OR } f_2 = \text{COMAND}(f_1, f_2) \bigcup f_2 \quad (17)$$

The AND operation between the block represented images f_1 and f_2 is implemented according to the formula:

$$f_1 \text{ AND } f_2 = \bigcup_{i \in [0,k_1-1]} \bigcup_{j \in [0,k_2-1]} (b_{i,f_1} \text{ AND } b_{j,f_2}) \quad (18)$$

where b_{m,f_n} denotes the m-th block of the n-th image. The operation AND between the two blocks results to a new block, defined by its coordinates as follows:

$$b_{i,f_1} \text{ AND } b_{j,f_2} = \big(\max(x_{1,b_{i,f_1}}, x_{1,b_{j,f_2}}), \min(x_{2,b_{i,f_1}}, x_{2,b_{j,f_2}}), \max(y_{1,b_{i,f_1}}, y_{1,b_{j,f_2}}), \min(y_{2,b_{i,f_1}}, y_{2,b_{j,f_2}}) \big) \quad (19)$$

The XOR operation between the blocks b_0, b_1 is implemented as:

$$b_0 \text{ XOR } b_1 = \text{COMAND}(b_0, b_1) \bigcup \text{COMAND}(b_1, b_0) \quad (20)$$

The execution of the XOR operation between two images, is the extension of the execution of the XOR operation between two blocks and is implemented according to the formula:

$$f_1 \text{ XOR } f_2 = \text{COMAND}(f_1, f_2) \bigcup \text{COMAND}(f_2, f_1) \quad (21)$$

The NOT operation in block represented images is implemented by the definition of an image f_Θ with the same width and height as the input image, containing only one block Θ that covers all the image. The execution of the operation COMAND(Θ, f) results to the inverse of the image f, i.e.

$$\text{NOT}(f) = \text{COMAND}(f_\Theta, f) \quad (22)$$

VI. APPLICATION TO EDGE EXTRACTION AND ENHANCEMENT

The edge enhancement and extraction in an image S with CL filters may be achieved by extracting at first the erosion S_B^E of the image S and then by subtracting the eroded image from the original. The above operation is faster using CL filters than topological min/max operations. Various combinations may be used in order to obtain the edges of the image, such as $S_B^D - S$, $S - S_B^E$, $S_B^D - S_B^E$.

Fig. 3, shows an example concerning a Magnetic Resonance (MR) image and the output of the gradient and two CL filters. Fig. 3(a) shows the original MR image. Fig. 3(b) shows the output of the

gradient operation to the MR image. The gradient operator, is used for edge extraction tasks. Fig. 3(c) shows the output of the following CL filter

$$f(i,j) = s(i,j) \text{ CAND } \{\text{CNOT } [s(i-1,j) \text{ CAND } s(i+1,j) \text{ CAND } s(i,j+1) \text{ CAND } s(i,j-1)]\} \quad (23)$$

that uses a 3x3 rhombus shaped structuring element. The use of a similar 5x5 rhombus shaped structuring element, produces bolder edges, as it is shown in Fig. 3(d).

It is clear from Fig. 3, that the use of CL filters produces better results, in comparison with the gradient operator. Also, the use of block represented images for the execution of the logic operations required by the CL filters, reduces the computational time.

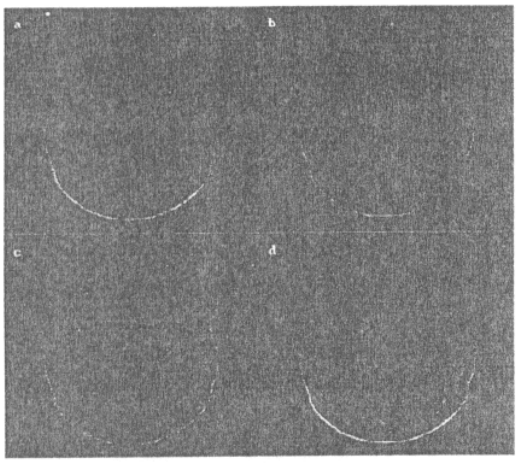

Fig. 3 The application of edge extraction and enhancement techniques to a MR image. (a) The original MR image. (b) The gradient of the original image. (c) The CL filter with a 3x3 structuring element. (d). The CL filter with a 5x5 structuring element.

ACKNOWLEDGMENT

This work is partial supported by the European Commision under the Human Capital and Mobility programme, with No. CHRX-CT94-0432.

REFERENCES

[1] K. Tsirikolias and B.G. Mertzios, "Logic Filters in Image Processing," *Proceedings of the International Conference on Digital Signal Processing*, pp. 285-287, Florence, Italy, Sept. 4-6, 1991.

[2] B.G. Mertzios and K. Tsirikolias, "Coordinate logic filters in image processing", *IEEE Winter Workshop on Nonlinear Digital Signal Processing*, Tampere, Finland, Jan. 17-20, 1993.

[3] K. Tsirikolias and B.G. Mertzios, "Edge Extraction and Enhancement Using Coordinate Logic Filters", *Proceedings of the International Conference on Image Processing: Theory and Applications*, pp. 251- 254, San Remo, Italy, June 14-16, 1993.

[4] D.L. Dietmeyer, *Logic Design of Digital Systems*, Allyn and Bacon Inc., 1971.

[5] J. Serra and L. Vincent, "An overview of morphological filtering", *Circuits Systems Signal Processing*, vol. 11, No 1, pp. 47-108, 1992

[6] P. Maragos and R. W. Schafer, "Morphological systems for multidimensional signal processing", *Proc. IEEE*, vol. 78, No. 4, pp. 690-710, April 1990.

[7] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, 1982.

[8] Y. Nakagawa and A. Rosenfeld, "A note on the use of local min and max operations in digital picture processing", *IEEE Trans. on Systems, Man, and Cybernetics*, vol. SMC-8, No. 8, pp. 632-635, August 1978.

[9] I. Pitas and A.N. Venetsanopoulos, *Nonlinear Digital Filters: Principles and Applications*, Boston: Kluwer Academic Publishers, 1990.

[10] J.H. Lin and E. J. Coyle, "Minimum mean absolute error estimation over the class of generalized stack filters", *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-38, No.4, pp. 663-678, April 1990.

[11] H. Samet, "The quadtree and related hierarchical data structures", *Computing Survey*, vol. 16, No. 2, pp. 187-260, 1984.

[12] H. Freeman, "Computer processing of line drawings", *ACM Computing Surveys*, vol. 6, pp. 57-97, 1974.

[13] D.W. Paglieroni and A.K. Jain, "Control point transfroms for shape representation and measurement", *Computer Vision, Graphics and Image Processing*, vol. 42, pp. 87-111, 1988.

[14] R.L. Kashyap and R. Chellappa, "Stochastic models for closed boundary analysis: Representation and reconstruction", *IEEE Trans. Information Theory*, vol. 27, pp. 627-637, 1981.

[15] J. Piper, "Efficient implementation of skeletonisation using interval coding", *Pattern Recognition Letters*, vol. 3, pp. 389-397, 1985.

[16] I.M. Spiliotis and B.G. Mertzios, "Real-Time computation of statistical moments on binary images using block representation", *4th International Workshop on Time-Varying Image Processing and Moving Object Recognition*, pp. 27-34, Florence, Italy, June 10-11, 1993.

[17] I.M. Spiliotis, D.A. Mitzias and B.G.Mertzios, "A skeleton-based hierarchical system for learning and recognition", *Proceedings of MTNS 93, International Symposium on the Mathematical Theory of Networks and Systems*, pp. 873-878, Regensburg, Germany,

[18] I.M. Spiliotis and B.G. Mertzios, "Real-time computation of two-dimensional moments on binary images using image block representation", *IEEE Trans. on Image Processing*. Submitted for publication.

[19] I.M. Spiliotis and B.G. Mertzios, "Pattern analysis on binary images using image block representation", *Pattern Recognition*. Submitted for publication.