Non-linear Edge Detectors Based on the Majority Gate

A. Gasteratos, I. Andreadis and Ph. Tsalides
Laboratory of Electronics
Section of Electronics and Information Systems Technology
Department of Electrical and Computer Engineering
Democritus University of Thrace
GR-671 00 Xanthi, Greece
E-mail: {agaster, ioannis}@orfeas.ee.duth.gr

Abstract: **A new technique for implementation of morphological edge detectors, including median prefiltering is presented in this paper. The proposed technique uses a bit-serial algorithm based on the majority gate. Several morphological edge detectors are studied and experimental results are also presented.**

Keywords: **Edge Detection , Rank Order Filters Mathematical Morphology.**

1. Introduction

An edge is the region of the image where the pixel intensity changes rapidly with respect to spatial changes. Such regions often represent the object boundary or different parts of the same object and, therefore, they carry information useful for image segmentation and recognition. The edge detection problem has been extensively studied by many researchers and a number of edge detection techniques have been reported [2-5, 7].

A major problem in edge detection is noise suppression. An edge detector must be robust in the presence of noise and able to discriminate between edges and regions corrupted by noise. Thus, the first stage of an edge detector is usually a noise suppression filter, such as a mean or a median filter. If the noise has Gaussian distribution, then a linear filter such as a mean filter gives good results. However, for other types of noise, such as impulsive or heavily-tailed noise, linear filters do not perform well. In such cases non-linear filters should be adopted. One such example is the median filter, which preserves edges and, therefore, valuable information needed for edge detection is maintained.

Edge detectors can be classified into two categories: linear and non-linear. Non-linear edge detectors are in general the difference of two non-linear filters. Such detectors are range filters and morphological edge detectors. In this paper, the implementation of non-linear edge detectors which employ median prefiltering and morphological edge detection is presented. The proposed architecture is based on the majority gate algorithm. This algorithm results into a single hardware module capable of handling both rank order and morphological filtering in real-time. Its simple structure along with the pipeline implementation allow real-time processing operation, and, thus the module will be useful for time critical-applications. Real-time video signal processing is a demanding task and usually requires the use of specific hardware.

2. Morphological Edge Detection

Mathematical morphology has been used for solving many image processing problems [6]. The two basic morphological operations are dilation and erosion. From these two all the other morphological transforms can be extracted. These transforms take advantage of the presence of the structuring element. In terms of grey-scale morphology 2-dimensional dilation is defined by:

$$d(x,y)=(f\oplus k)(x,y)=\max_{\substack{(i,j)\in \mathrm{K}\\(x,y)-(i,j)\in \mathrm{F}}}\{f(x-i,y-j)+k(i,j)\} \quad (1)$$

where x, y are spatial co-ordinates, $f:\ \mathrm{F}\rightarrow \mathrm{E}$ is the image under process and $k:\ \mathrm{K}\rightarrow \mathrm{E}$ is the structuring element; E is the Euclidean space of dimension 1 and $\mathrm{F},\mathrm{K}\subseteq \mathrm{E}\times \mathrm{E}$.

Similarly, erosion is defined by:

$$e(x,y)=(f\Theta k)(x,y)=\min_{(i,j)\in \mathrm{K}}\{f(x+i,y+j)-k(i,j)\} \quad (2)$$

Morphological edge detectors are a class of morphological transforms. The simplest members of this class are dilation residue and the erosion residue edge detectors. These are defined by [3]:

$$G_d(x,y)=d(x,y)-f(x,y) \quad (3)$$

and

$$G_e(x,y)=f(x,y)-e(x,y) \quad (4)$$

respectively.

The above defined edge detectors highlight rapid grey-level transitions in the image. If the structuring elements are symmetrical these operations tend not to depend on edge directionality, as opposed to gradient edge detectors. However, the exact object size is not always found. Let us consider the effect of dilation operation on an object brighter than its background. Dilation expands the object and, therefore, dilation residue outputs the external edges of the object. Erosion shrinks the object and, therefore, erosion residue outputs the internal edges of the object. Now, if the object is darker than its background, dilation residue outputs the internal edges of the object and erosion residue outputs the external edges of the object. From the previous discussion it becomes clear that these edge detectors are biased by the brightness/darkness of the objects. This bias can be attenuated by introducing an edge detector, which combines the dilation residue and erosion residue edge detectors, using pixel-wise maximum, minimum, or sum, which are defined as follows:

$$G_{\max}(x,y)=\max[G_d(x,y)+G_e(x,y)] \quad (5)$$

$$G_{\min}(x,y)=\min[G_d(x,y)+G_e(x,y)] \quad (6)$$

and

$$G_{de}(x,y)=G_d(x,y)+G_e(x,y)=d(x,y)-e(x,y) \quad (7)$$

respectively.

The previously defined edge detectors cannot detect ideal step edges ($G_{\max}$ and G_{de}) and they are also sensitive to impulse noise ($G_{\max}$ and $G_{\min}$). However, ideal step edges do not occur in real images. Also, impulse noise can be suppressed by median filtering. Therefore, one or more stages of median prefiltering, depending on the level of noise, followed by any of the above edge detectors results into efficient edge detector schemes.

3. Experimental Results

In order to state the performance of the previously described edge detectors several images have been obtained through a commercial frame grabber. Figure 1(a) shows the original input image. Numbering is done from left to right The results of G_{e}, G_{d}, G_{de}, $G_{\min}$ and $G_{\max}$ edge detectors when these are applied to the input image are shown in Figures 1(b), (c), (d), (e) and (f), respectively. The same symmetrical 3x3-pixel structuring element has been used in all experiments. The results of erosion and dilation operators have been limited, when required, in the range [0 ... 255].

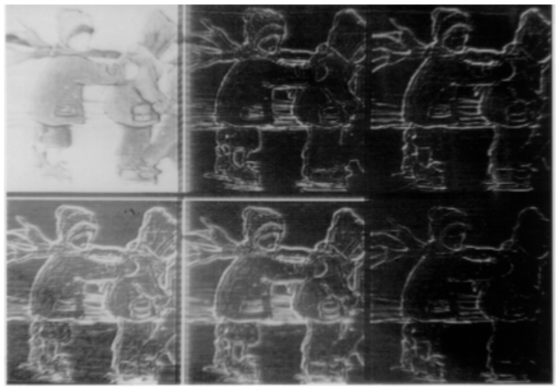

Figure 1. (a) Image, (b) G_{e} image, (c) G_{d} image, (d) G_{de} image, (e) $G_{\min}$ image and (f) $G_{\max}$ image.

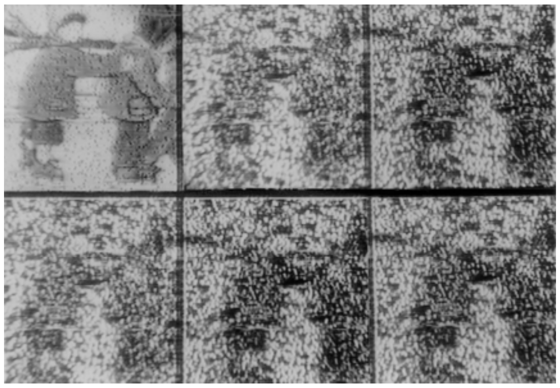

Figure 2. (a) Impulse noise corrupted image, (b) G_e image, (c) G_d image, (d) G_{de} image, (e) G_{min} image and (f) G_{max} image.

Figure 2 (a) shows the image of Figure 1(a), corrupted by additive impulsive noise. Figures 2(b), (c), (d), (e) and (f) present the results of G_e, G_d, G_{de}, G_{min} and G_{max} edge detectors, respectively, on the image of Figure 2(a). As it was expected noise does degrade the performance of these edge detectors. Figure 3 (a) shows the output of median filtering on the image of Figure 2(a), using a 3x3-pixel mask. The effect of median filtering results into satisfactory edge detection. This is illustrated in Figures 3(b), (c), (d), (e) and (f), which correspond to the outputs of G_e, G_d, G_{de}, G_{min} and G_{max} edge detectors, respectively.

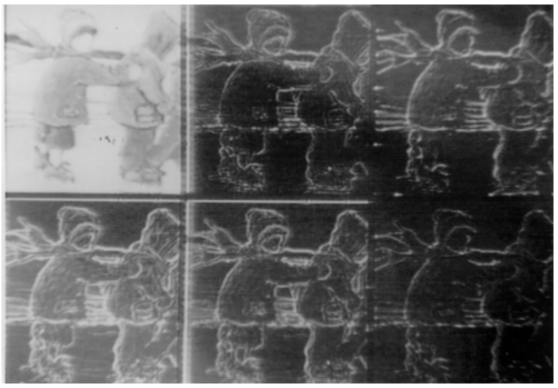

Figure 3. (a) Median filtered image, (b) G_e image, (c) G_d image, (d) G_{de} image, (e) G_{min} image and (f) G_{max} image.

4. Hardware Implementation of Edge Detectors

The implementation of median filters and morphological operators has been unified in an efficient algorithm [1]. This algorithm allows the implementation of a single hardware module capable of computing any order statistic (order statistic hardware module - o.s.h.m.), such as median, maximum and minimum. Maximum and minimum are used for the computation of dilation and erosion, respectively(eqns. (1) and (2)). The edge detectors which have been previously described can be implemented using: *n* o.s.h.m.s for *n* successive median prefiltering interactions, one o.s.h.m. and an adder for the dilation operation, one o.s.h.m. and a subtractor for the operation of erosion and, finally, an adder for G_{de}, or an o.s.h.m. for G_{max}or G_{min}. The use of the same o.s.h.m. in all different parts of the edge detection circuitry, results into a more controllable structure. As an example, the block diagram of G_{min} is shown in Figure 4.

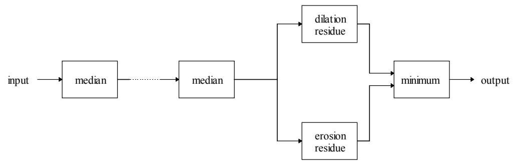

Figure 4. Block diagram of G_{min} edge detector with median prefiltering.

The algorithm, which is used for the implementation of the o.s.h.m. extracts any order statistic of W numbers in a data window (W is a positive, odd integer number), by computing the median value of 2W-1 numbers. The extra W-1 numbers are dummy numbers. The key-concept here is that by having a method to compute the median value of an expanded data window and by being able to control the dummy numbers, any order statistic of the original data window can be determined. Figures 5(a), (b) and (c) illustrate this concept. The dummy numbers are used for the computation of median (3rd order statistic), minimum (1st order statistic) and maximum (5th order statistic) values, respectively. The bold window contains five numbers $\alpha_{(1)}$, $\alpha_{(2)}$, ...$\alpha_{(5)}$ in an ascending order (the subscript in the parentheses denotes the rank). The larger window contains nine numbers also in an ascending order. By controlling the number of the dummy numbers which are pushed to the top and to the bottom, any order statistic r of the numbers α_i, can be obtained.

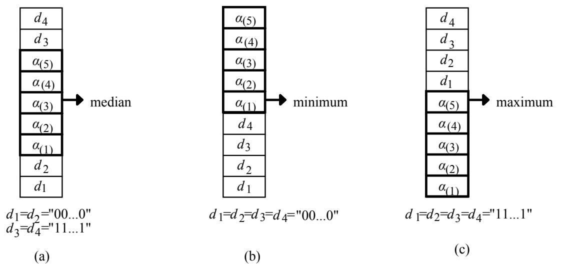

Figure 5. Illustration of dummy inputs use.

As an illustrative example, for the computation of the median value of the 2W-1 numbers the following notations and definitions are used:

Suppose that the median value of numbers α_i is:

m=med(α_1 , α_2 ,...α_W) (8)

α_i numbers are represented in binary form of k-bit resolution. Suppose that $b_{i,j}$ is the j th bit of the binary representation of x_i , then :

$$a_i = \sum_{j=1}^{k} b_{i,j} 2^{k-j} \tag{9}$$

Also, suppose that o_j is the j th bit of the binary representation of m, then:

$$m= \sum_{j=1}^{k} o_j 2^{k-j} \tag{10}$$

The following flags and intermediate signals are defined:

$r_{i,j}$ is the rejecting flag, which indicates whether number α_i remains within the subset of the candidate numbers to be the median, in the j th step of the algorithm. When $r_{i,j}$ is "1" number α_i remains within the subset, otherwise α_i is rejected. Once r_i is set to "0" it remains constant and the remaining bits b_i , are not taken into account.

$t_{i,j}$ is the setting flag, which replaces the $b_{i,j}$ bits of the rejected numbers in the majority selection process. It is set to the complementary value of the previous output bit o_{j-1} , when the number has been rejected. In this way, the number which has been rejected is pushed away from the median value. If $r_{i,j}$ has not been changed to "0", then $t_{i,j}$ is in a "don't care" state.

$i_{i,j}$ is an intermediate signal, which is either $b_{i,j}$ if α_i has not been rejected, or $t_{i,j}$ if α_i has been rejected. If the majority of $i_{i,j}$ is "1" then the output bit o_j is "1" , otherwise it is "0".

The median value computation procedure follows. The MSBs of the numbers in the list are first processed, by dedicated Processing Elements (PEs). The realisation of the proposed (PE) is based on eqns. (4), (5) and (7). The circuit diagram of this PE is shown in Figure 6. The other bits are then processed sequentially by identical PEs until the less significant bits (LSBs) are reached. Initially, the rejecting flag signals $r_{i,\ 1}$ are set to "1", since all the numbers are candidates to be the median value. The setting flags $t_{i,1}$ are in a "don't care" state. If the majority of the MSBs $b_{i,1}$ is found to be "1", then the MSB of the output is o_1="1", otherwise o_1="0". In the following stage bits $b_{i,2}$ of the numbers which have MSBs complement to o_1 are rejected and are not taken into account for the majority selection. Instead, signals $t_{i,2}$, which correspond to the rejected numbers, are taken into account. The majority selection procedure continues in the next stages and the median value m is obtained by collecting bits o_j according to eqn. (10). The majority selection can be efficiently realised using a majority gate CMOS device. The hardware implementation of the o.s.h.m. is shown in Figure 7. This is a pipelined systolic array, the inputs to which are the 2W-1 numbers. Its simple structure along with the pipeline implementation allow real-time processing, and, thus such a module will be suitable for time critical applications.

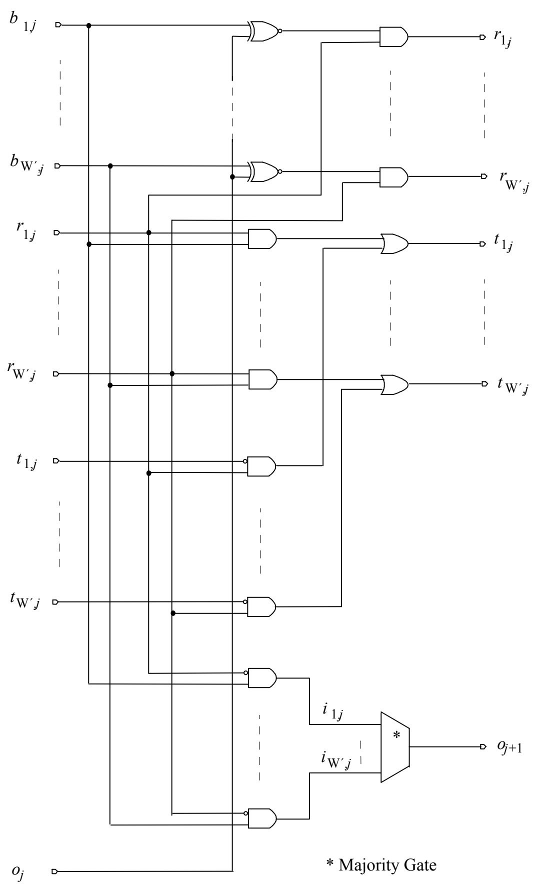

Figure 6. The basic processing element (PE) of the o.s.h.m..

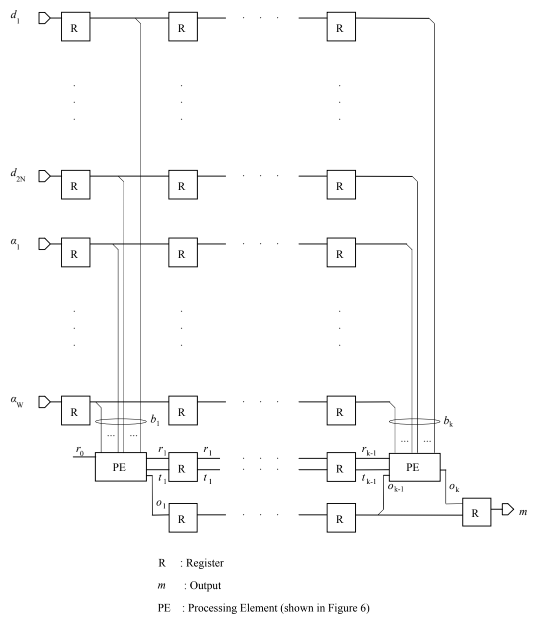

Figure 7. Block diagram of the o.s.h.m.

5. Conclusions

The implementation of morphological edge detectors, including median prefiltering, through a single hardware module has been presented in this paper. The proposed implementation is based on the majority gate. Using this single module various edge detectors such as G_{de}, $G_{\min}$ and $G_{\max}$ can be constructed. Furthermore, the use of a single building module in all different parts of the edge detection circuitry results into a more controllable structure.

References

1. Gasteratos, A., Andreadis, I. and Tsalides, Ph. (1997) Realization of Rank Order Filters Based on the Majority Gate. *Pattern Recognition* **30(9):** 1571-1576.
2. Hardie, R.C. and Boncelet, C.G. (1995) Gradient-based edge detection using non-linear edge enhancing prefilters. *IEEE Trans. on Imag. Proc.*, **4(11):** 1572-1577.
3. Lee, J.S.J., Haralick, R.M. and Sapiro, L.G. (1987) Morphological edge detection. *IEEE Trans. on Robotics and Automation*, **3(2):** 142-156.
4. Neuvo, Y., Heinonen, P. and Defee, I. (1987) Linear-median hybrid edge detectors. *IEEE Trans. on Circuits and Systems*, **CAS-34(11):** 1337-1343.
5. Pitas, I. and Venetsanopoulos, A.N. (1990) *Non-linear Digital Filters Principles and Applications,* Boston, Kluwer Academic Publicers.
6. Serra, J. (1982) *Image Analysis and Mathematical Morphology, Vol. I*, New York, Academic Press.
7. Yoo, J., Bouman, C.A., Delp, E.J. and Coyle, E.J. (1993) The non-linear prefiltering and difference of estimates approaches to edge detection: application of stack filters. *CVGIP : Graphical Models and Image Processing*, **2(2):** 140-159.