

Threshold Boolean Filtering with Cellular Neural Networks

Marco BALSÌ† Mauro ZACCARI‡

Abstract

Threshold Boolean Filters (TBF) [1] are used in a broad class of signal and image processing applications, especially in the presence of non-Gaussian noise, and in nonlinear feature extraction.

In this paper, we give techniques to implement TBFs in the Cellular Neural Network Universal Machine (CNN-UM) architecture [2]. In this way, we give a new solution for implementation of TBFs in a general purpose fully parallel architecture, and at the same time obtain a systematic procedure to generate "analogic" programs for the CNN-UM.

1 Introduction

Threshold Boolean Filters form a wide class of nonlinear filters that includes Stack Filters (SF) [3], Order Statistic (OS) filters [4,5], and morphological filters [6] as special cases. Such filters find application in removal of non-Gaussian noise and feature extraction [3,7,8,9], and are therefore used for many purposes in image processing.

Architectures for realization of TBFs have been proposed in the literature [1,3,10], generally as sequential processors exploiting the threshold decomposition property, to be defined below.

In this paper, we shall show that TBFs can be realized as "analogic" programs in the CNN-UM framework, so that a fully parallel analog implementation of TBFs is obtained. Moreover, design techniques for such filters [8,11] may be applied to the design of CNN-UM algorithms, so that for the first time an analogic program is obtained by a totally systematic procedure.

2 Threshold Boolean Filters

Threshold Boolean Filters are defined by the threshold decomposition property, i.e. the fact that the filtering

$$\mathbf{Y} = F(\mathbf{X}) = \sum_{l=1}^M f(T_l(\mathbf{X})) = \sum_{l=1}^M T_l(\mathbf{Y}) \quad (1)$$

In Eq. 1, \mathbf{X} is the signal being filtered, an array of samples within a suitable window. F represents the TBF; T_l is the operation of thresholding at the l -th of M levels, and f is a Boolean function. While in TBF literature signals are usually taken in the discrete range $[0, M]$, we shall consider here continuous range $[-1, 1]$, so that the level of threshold l is $2l/(M+1)-1$. Therefore, Eq. (1) becomes

$$\mathbf{Y} = -1 + v \sum_{l=1}^M f(T_{lv-1}(\mathbf{X})) \quad (1b)$$

where $v = 2/(M+1)$

A very important subclass of TBFs is that of stack filters, defined as those TBFs that have the stacking property, which is expressed as

$$\mathbf{x} \leq \mathbf{y} \Rightarrow f(\mathbf{x}) \leq f(\mathbf{y})$$

where a partial ordering has been defined among binary arrays by stating that $\mathbf{x} \leq \mathbf{y}$ iff each element of \mathbf{x} is less than or equal the same element of \mathbf{y} . The stacking property involves simplified computation and realization, because the summation in Eq. 1 can be replaced by a binary search for the threshold level where the transition between 0 and 1 output appears [3]. A necessary and sufficient condition for the stacking property to hold is that f can be represented by an expression that does not contain any complemented variable, i.e. f is a positive Boolean function. Morphological and OS filters are themselves SFs.

```
m1:=original_image;
xFILL(false,m3,dim);
REPEAT L:=1 TO M BY 1
  threshold_L(m1,m1,m2);
  xFILL(false,m2,dim);
  REPEAT i:=1 TO K BY 1
    prod_i(m2,m2,m2);
  ENDREPEAT;
  m3:=m3+m2*norm;
ENDREPEAT; /* here m3 holds result */
```

Figure 1

Analogic program for realization of TBFs (solution based on linear templates). Constant false is -1; norm is a normalization constant whose value is $2/(\text{no. of threshold levels } M)$; dim is no. of pixels of the image being processed; K is no. of products in Boolean function.

† Interuniversity Center for Research on Cognitive Processing in Natural and Artificial Systems (E.Co.N.A.), Rome, Italy, and

Dipartimento di Ingegneria Elettronica, Università "La Sapienza", via Eudossiana 18 Rome, Italy I-00184.

<http://tce.ing.uniroma1.it/balsi.html>

‡ Dipartimento di Ingegneria Elettronica (as above)

operation can be performed separately on an array of binary signals obtained by thresholding the original signal at several levels:

```

m1:=original_image;
xFILL(false,m4,dim);
REPEAT i:=1 TO P BY 1
  min_i(m1,m1,m2);
  max_i(m1,m1,m3);
  m2:=m2-m3;
  pw1(m2,m2,m2);
  m4:=m4+m2;
ENDREPEAT; /* here m4 holds result */

```

Figure 2

Analogic program for realization of TBFs (solution based on nonlinear difference-controlled templates). Here P is the no. of products in the SOMEPE representation of the i-th such filter.

A TBF can be always obtained as a linear combination of SFs, so that any technique for implementation of the latter can be used for TBFs with minor modifications.

Another relevant class is that of linearly separable (LS) TBFs, which are obtained with an LS function f (linear threshold). Such filters also enjoy simpler design and realization.

A useful alternative representation (called “multilevel” [1]) of TBFs is based on expressing function f in the form of sum of mutually exclusive products (SOMEPE). In this case, let m_i denote the minimum of those input samples that appear uncomplemented in the i-th product, and M_i the maximum of the complemented ones. Then, with the normalization used,

$$F(\mathbf{X}) = \sum_{i=1}^P \max(-1, m_i(\mathbf{X}) - M_i(\mathbf{X})). \quad (2)$$

SFs also admit another multilevel representation, based on the fact that positive Boolean functions commute with thresholding, as logical AND and OR are changed into minimum and maximum respectively. Therefore, when f is written as sum of positive products, the filter can also be written as

$$F(\mathbf{X}) = \max_i [m_i(\mathbf{X})]. \quad (3)$$

It is appropriate to notice that Eq. 3 is not a particular case of Eq. 2, because they are based on a different representation of f .

Several algorithms have been proposed for design of various classes of TBFs. In the case of SFs, Lin & Kim [11] proposed an algorithm based on adaptation of the truth table of f over examples, with the purpose of minimizing mean absolute error (MAE). In the case of LS TBFs, Lee & Lee [1] compute directly the weight vector that minimizes MAE over the examples.

3 Implementing TBFs as CNN-UM analogic programs

We have developed two basic solutions for realizing TBFs in the CNN-UM framework. The first solution employs linear templates, while the second

```

m1:=original_image;
min_1(m1,m1,m2,t,0);
REPEAT i:=2 TO P BY 1
  min_i(m1,m1,m3);
  loc_max(m2,m3,m2);
ENDREPEAT; /* here m2 holds result */

```

Figure 3

Analogic program for realization of SFs (alternative solution based on Eq. 3)

employs difference-controlled nonlinear templates [12] but is significantly faster.

The first solution is based on sum-of-products representation of Boolean function f . Realization of a generic Boolean function in CNNs is possible using the technique developed by Galias [13]. Therefore, based on the definition of TBF, a simple analogic program can be written based on two cycles (Fig. 1-relevant lines of the program written in a simplified Alpha language [14]). In the inner cycle over i , each product in the expression of f is computed (template prod_i - arguments are input, initial state, output), while in the outer cycle over L , current thresholded image is obtained using template threshold_L (thresholding from [15]), function f computed by the inner cycle, and the result summed in an accumulator after proper normalization.

If the filter being implemented is a stack filter, then the outer cycle may be broken when current result of prod_i is a uniform image at -1 value (such a check is a primitive of CNN-UM computing).

It is to be noted that templates prod_i are switched cyclically among a relatively small number K , and could be stored in local APR if available, while templates threshold_L are in fact always the same template, where only the value of L is changed. Actual implementation of such operation depends on chip architecture, and might be done locally.

The second solution we have devised is based on the multilevel representation of TBFs. The maximum and minimum filters are in fact particular order statistic filters, that are available as nonlinear difference-controlled templates in the CNN framework [12,16]. Therefore, the program shown in Fig. 2 can be straightforwardly obtained.

Templates min_i and max_i have the same form (cf. [12]) rank order templates:

$$A=1, \hat{D} = \begin{bmatrix} d & d & d \\ d & 0 & d \\ d & d & d \end{bmatrix}, d = Q_{kl} \text{sign}(\Delta v_{ux}), I$$

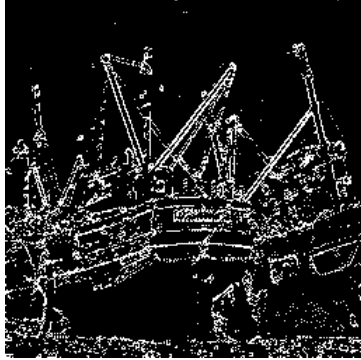
where in the case of min_i $Q_{kl} = 1$ for those positions corresponding to uncomplemented variables in the i-th product in the SOMEPE expression of f , else $Q_{kl} = 0$; and $I = N_u - 1$, with N_u number of the said uncomplemented variables. For max_i , complemented variables are considered, and $I = 1 - N_c$.



(a)



(b)



(c)

Figure 4

- (a) Lena image, corrupted with impulsive noise;
 (b) Edges extracted by designed CNN-UM TBF;
 (c) same TBF applied to "boats" image

It is to be noted that the first solution involves a cycle over the number of threshold levels, which will normally be a large number, while the cycle in the second solution is quite short.

For this reason, the latter should be preferred when nonlinear connections are available, while the first is available even using existing CNN hardware.

Based on Eq. 3, in the case of SFs, it is also possible to use the algorithm outlined in Fig. 3. Template loc_max , that yields the maximum between local input and initial state, is obtained as follows:

$$A = 1, \hat{D} = [d], I = 0,$$

where

$$d = \begin{cases} \Delta v_{ux} & \text{if } \Delta v_{ux} > 0 \\ 0 & \text{else} \end{cases}.$$

Which solution is to be preferred depends on the number of products in the two representations of the filter considered.

4 Examples

In order to test application of the techniques outlined in the paper, we considered two examples.

The first example is edge detection in the presence of impulsive noise. In this case, in order to check the methods, we made the design with the purpose of obtaining a known TBF solution of the problem. Such a solution is based on the idea of labeling as edges those locations where the difference between dilated and eroded images exceeds a given threshold [9].

We designed the stack filters for erosion and dilation by applying Lin & Kim's algorithm on a 40×40 pixel section comprising the right eye of Lena's image corrupted by 10% impulsive noise with magnitude 200 (on a range of 256).

The filter obtained was mapped to the CNN and tested on the whole Lena and on different images such as "boats". Results (Fig. 4) were satisfactory (especially as only the first neighborhood is used, and noise is quite large), and show that good generalization is obtained.

We then considered a real-life classification problem, namely, segmentation of a sonar image of sea bed (335×568 pixel, 256 gray levels).

In this case, we trained an LS TBF to classify areas of the image into three classes, using a very small portion (85×64 pixels) as example, by manually segmenting it.

It is apparent, that the designed filter (operating on a 5×5 neighborhood), when applied to the whole image gives a good enhancing. The worst effect is obtained in those areas of the image that are not well represented in the example.

5 Conclusions

It has been shown that TBFs can be straightforwardly mapped into the CNN-UM architecture by means of a systematic design procedure. When CNN-UM processors are available, such implementation will permit considerable speed and flexibility of operation. Moreover, design techniques developed for TBFs can be used to obtain new analogic procedures for solution of practical signal and image processing problems on the CNN-UM.

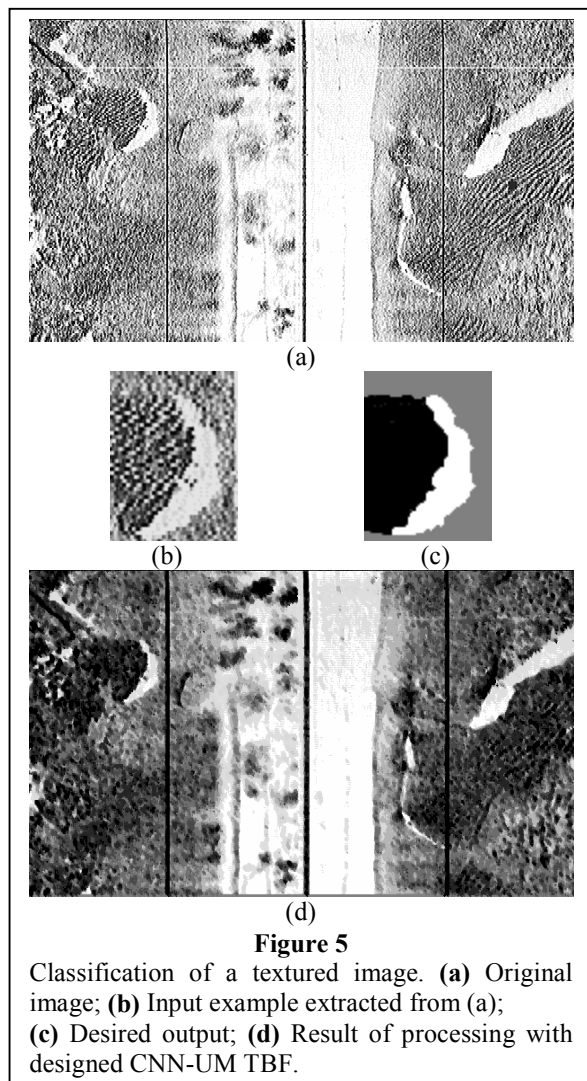


Figure 5

Classification of a textured image. (a) Original image; (b) Input example extracted from (a); (c) Desired output; (d) Result of processing with designed CNN-UM TBF.

Acknowledgements

Images in Fig. 5 (a) and (d) have been provided by courtesy of the SACLANT Undersea Research Center in the framework of the EU funded project ISACS, contract MAS3-CT95-0046, and used within the joint collaboration between "La Sapienza" University and DSEA - University of Pisa, Italy.

References

- [1] Lee, K.D., Lee, Y.H., "Threshold Boolean Filters", *IEEE Trans. on Sig. Proc.*, vol. 42, 1994, pp. 2022-2036.
- [2] Roska, T., Chua, L.O., "The CNN Universal Machine: an Analogic Array Computer", *IEEE Trans. on Circ. Syst.*, part II, vol. 40, 1993, pp. 163-173.
- [3] Wendt, P.D., Coyle, E.J., Gallagher, N.C., "Stack Filters", *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. 34, 1986, pp. 898-911.
- [4] Yu, P.-T., Liao, W.-H., "Weighted Order Statistics Filters - Their Classification, Some Properties, and Conversion Algorithm", *IEEE Trans. on Sig. Proc.*, vol. 42, 1994, pp. 2678-2691.
- [5] Yli-Harja, O., Astola, J., Neuvo, Y., "Analysis of the properties of Median and Weighted Median Filters Using Threshold Logic and Stack Filter Representation", *IEEE Trans. on Sig. Proc.*, vol. 39, 1991, pp. 395-410.
- [6] Maragos, P., Schafer, R.W., "Morphological Filters - Part I & II", *IEEE Trans. Acoustics, Speech, and Sig. Proc.*, vol. 35, 1987, pp. 1153-1183.
- [7] Coyle, E.J., Lin, J.-H., Gabbouj, M., "Optimal Stack Filtering and the Estimation and Structural Approaches to Image Processing", *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. 37, 1989, pp. 2037-2066.
- [8] Yin, L., Astola, J.T., Neuvo, Y.A., "Adaptive Stack Filtering with Application to Image Processing", *IEEE Trans. Sig. Proc.*, vol. 41, 1993, pp. 162-184.
- [9] Yoo, J., Coyle, E.J., Bouman, C.A., "Dual Stack Filters and the Modified Difference of Estimates Approach to Edge Detection", *IEEE Trans. Image Proc.*, vol. 6, 1997, pp. 1634-1645.
- [10] Fitch, J.P., "Software and VLSI Algorithms for Generalized Ranked Order Filtering", *IEEE Trans. on Circ. Syst.*, vol. 34, 1987, pp. 553-559.
- [11] Lin, J.-H., Kim, Y.-T., "Fast Algorithms for Training Stack Filters", *IEEE Trans. Sig. Proc.* vol. 42, 1994, pp. 772-781.
- [12] Rekeczky, Cs., Roska, T., Ushida, A., "CNN-Based Difference-Controlled Adaptive Non-Linear Image Filters", *int. j. circ. th. appl.*, vol. 26, 1998, pp. 375-423.
- [13] Galias, Z., "Designing Cellular Neural Networks for the Evaluation of Local Boolean Functions", *IEEE Trans. on Circ. Syst.*, II, vol. 40, 1993, pp. 219-223.
- [14] Zöld, S., "CNN Alpha Language and Compiler - version 1.1", *Hungarian Academy of Sciences, Computer and Automation Institute (MTA-SzTAKI), rep. DNS-10-1997, Budapest, Hungary, 1997.*
- [15] Roska, T., Kék, L., "Analogic CNN Program Library - vers. 6.0", *MTA-SzTAKI, rep. DNS-5-1994, Budapest, Hungary, 1994.*
- [16] Shi, B.E., "Order Statistic Filtering with Cellular Neural Networks", *Proc. of Third IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA-94), Rome, Dec. 18-21, 1994*, pp. 441-444.