An Adaptive Hybrid Morphological Method for Designing Translation Invariant Morphological Operators

Ricardo de A. Araújo, *Student Member, IEEE*, and Robson P. de Sousa, *Member, IEEE*

Abstract— **In this paper an adaptive hybrid morphological method is presented for designing translation invariant morphological operators via Matheron Decomposition and via Banon and Barrera Decomposition. It consists of a hybrid model composed of a Modular Morphological Neural Network (MMNN) and an evolutionary search mechanism: the adaptive Genetic Algorithm (GA) (adaptive rates in genetic operators). The proposed method searches for initial weights, architecture and number of modules in the MMNN; then each element of the adaptive GA population is trained via the Back Propagation (BP) algorithm. Also, it is presented and experimental investigation with the proposed method using a relevant application in image processing, the restoration of noisy images, where it verifies that the method proposed herein allows seamless and efficient design of translation invariant morphological operators of either increasing or non-increasing types, and the results are discussed and compared, in terms of Noise to Signal Ratio (NSR), with the previously methods proposed in literature, showing the robustness of the proposed method.**

Keywords— **Translation Invariant Morphological Operators, Morphological Neural Networks, Evolutionary Computation, Hybrid Systems, Image Restoration.**

I. Introduction

Mathematical Morphology (MM) represents an important class in image processing. It uses a nonlinear focus based on geometric analysis through using structuring elements, which are small patterns that operates in spatial domain and extract information about the geometric forms in the image [1], [2]. The MM was created around 1960, being developed by Matheron [3] and Serra [4] with goal of processing boolean images. Sternberg [5] and Serra [4] extended such concept for gray scale image processing. The MM formal definition, in terms of lattice algebra, was presented by Heijmans [6]. An important result for MM was presented by Banon and Barrera [7], the decomposition theorem of translation invariant operators, which guarantees that every translation invariant operator, not necessary increasing, may be decomposed by a combination of basic operators: dilation, erosion, anti-dilation and anti-erosion. The Banon and Barrera [7] theorem result is a generalization of the Matheron canonic decomposition theorem [3] for increasing translation invariant operators.

Ricardo de A. Araújo is with Center for Informatics, Federal University of Pernambuco, Av. Prof. Luiz Freire, s/n, CDU, 50732-970, Recife - PE - Brazil. Email: raa@cin.ufpe.br.

Robson P. de Sousa is with Center of Science and Technology, State University of Paraíba, 58109-790, Campina Grande - PB - Brazil. Email: sousarob@yahoo.com.br.

The design of translation invariant operators is a relevant problem in mathematical morphology, with applications in image processing, like image restoration, edge extraction and object recognition. Many works have focused on the design of this kind of operators. Yang and Maragos [8] designed operators (min-max classifiers) according to Matheron decomposition [3] by using the mean square error for minimizing the cost function. Dougherty and Loce [9] designed sub-optimal operators satisfying Matheron theorem [3] for binary image processing. Davidson and Hummer [10] used Morphological Neural Networks (MNNs) for designing morphological filters, differing from the classical neural networks [11] in the sense that the computation in each node of the MNN is carried out by simple morphological operators in the context of Algebra of Images [12]. Herwing and Shalkoff [13] presented a MNN with learning based on the delta rule for designing filter for binary images.

Pessoa and Maragos [14] proposed a general class of neural network architectures involving morphological/rank/linear operators, from which traditional multi-layer perceptrons and morphological/rank networks can be designed using the back propagation algorithm. Harvey and Marshall [15] used Simple Genetic Algorithm (SGA) for designing morphological filters for gray level images. Oliveira [16] generalized the work of Harvey and Marshall [15] by implementing Banon and Barrera decomposition [7] via SGA for operators non necessarily linear. Based on [14], Sousa [1], [2] presented particular network architectures, referred to as Modular Morphological Neural Networks (MMNNs), via the Matheron decomposition theorem [3] and the more general Banon and Barrera decomposition theorem [7]. The MMNN training in [1], [2] can be via the SGA or via the back propagation (BP) algorithm. Based on [1], [2], Araújo et al. [17] presented an adaptive evolutionary method for designing translation invariant morphological operators applied to image restoration.

This paper presents an adaptive hybrid morphological method for designing translation invariant operators via Matheron Decomposition and via Banon and Barrera Decomposition for image restoration of images corrupted by salt and pepper noise. It consists of a hybrid model composed of an MMNN [1], [2] and an adaptive Genetic Algorithm (GA) [18], which may be regarded as an improvement of the methodology described in [17] as follows: (a) the GA is adaptive for faster convergence; (b) each element of the adaptive GA population represents an MMNN; and (c) an adaptive GA is used to search for the initial weights, architecture and number of modules of the MMNN, wherein each element of the adaptive GA population is trained via the BP algorithm.

II. Background

A. Mathematical Morphology

Mathematical Morphology represents an important class of nonlinear signal processing systems, which aims to quantitatively describe the geometrical structure of a signal using structuring elements. The following equations are used in MMNN for designing translation invariant operators [1], [2]:

$$\text{Dilation: } \delta_k = max\Big(\vec{x} + \vec{a}_k\Big); \tag{1}$$

$$\text{Erosion: } \epsilon_k = min\Big(\vec{x} - \vec{a}_k\Big); \tag{2}$$

$$\text{Anti-Dilation: } \delta_k^a = 1 - min\left(\vec{x} - \vec{b}_k\right); \tag{3}$$

$$\text{Anti-Erosion: } \epsilon_k^a = 1 - max\left(\vec{x} + \vec{b}_k\right), \tag{4}$$

where $\vec{x}$ is the input signal and $\vec{a}_k$ and $\vec{b}_k$ represent the structuring element (terms $\vec{b}_k$ represent the reflection of the complement of the structuring elements of anti-dilation or anti-erosion).

B. Adaptive Genetic Algorithm

The adaptive GA of Mitsuo and Cheng [18] differs from SGA by using adaptive methods applied to crossover and mutation operators. The method adopted in the present work is the deterministic adaption [18] and consists in modifying the operators rate according to a pre-stablished rule. The operators rates are gradually decreased in each population evolution. The following equation defines the rule adopted as the adaptive parameter in the rates of crossover and mutation:

$$Tx_a = Tx_i - (Tx_i - Tx_f) * \frac{g_a}{G}, \tag{5}$$

where Tx_a, Tx_i and Tx_f represent the current, initial and final rates. The terms G and g_a represent, respectively, the maximum number of generations and the current generation.

Figure 1 illustrates the adaptive GA procedure.

```
AdaptiveGeneticAlgorithmProcedure() {
  τ = 0;                    // τ: iteration number
  initialize P(τ);          // P(τ): population for iteration τ
  evaluate f(P(τ));         // f(·): fitness or cost function
  while ( not termination condition ) {
    τ = τ+1;
    select individuals parents pairs from P(τ-1);
    perform crossover operator with the selected
      individuals parents pairs;
    generate a new P(τ);
    perform the mutation operator with the new P(τ);
    evaluate f(P(τ));
    update the crossover and mutation rates according
      to Equation (5);
  }
}
```

Fig. 1. Adaptive genetic algorithm procedure.

According to Figure 1, the adaptive GA procedure starts with the creation of an individuals' population, or more specifically, the solutions set. Then, each individual is evaluated by a fitness function (or cost function), which is an heuristic function that guides the search for an optimal solution in state space. After evaluating the GA population, it's necessary to use some procedure to select the individual parent pairs, which will be used to perform the genetic operators (crossover and mutation). The next step is responsible to performing the crossover genetic operator. Usually, the crossover operator mixture the parents genes for exchanging genetic information from both, obtaining its offspring individuals. After crossover operator, all offspring individuals will be the new population, which contains relevant characteristics of all individuals parent pairs obtained in selection process. The next step is to mutate the new population. The mutation operator is responsible by the individual genes aleatory modification, allowing the population diversification and enabling GA to escape of local minima (or maxima) of the surface of the cost function (fitness). Finally, the new mutated population is evaluated and then the crossover and mutation rates are updated according to Equation 5. This procedure is repeated until a stop condition be reached.

III. MMNN Fundamentals

A. MMNN Architecture for the Matheron Decompositon

Sousa [1], [2] defined the MMNN for designing translation invariant operators that satisfy the Matheron decomposition theorem [3] for dilations as well as for erosions. The Matheron theorem [3] states that every increasing and translation invariant operator may be decomposed by a union of erosions or a intersection of dilations. Figure 2 presents the MMNN architecture for the Matheron decomposition [3] by dilations. The following equations define the MMNN architecture for the Matheron decomposition [3] via dilations according to this approach.

$$v_k = \delta_k = max\Big(\vec{x} + \vec{a}_k\Big), \tag{6}$$

where $\vec{x}$ is the input signal of the MMNN.

$$\text{Network output: } Y = min(\vec{v}), \tag{7}$$

where

$$\vec{v} = (v_1,\ v_2,\ \ldots\ , v_k). \tag{8}$$

The weight matrix, A, of the MMNN is defined by

$$A = (\vec{a}_1;\ \vec{a}_2;\ \ldots;\ \vec{a}_k), \tag{9}$$

where $\vec{a}_k \in \mathcal{R}^k$, $k = 1, 2, \ldots, N$, represent the MMNN weights (i.e. rows composed by structuring elements $\vec{a}_k$). Symbol $\wedge$ represents the minimum operation.

In a dual manner, the MMNN architecture for the Matheron decomposition [3] via erosions is defined by substituting dilations by erosions and symbol $\wedge$ by $\vee$, where $\vee$ represents the maximum operation. Figure 3 presents the MMNN architecture for the Matheron decomposition [3] by erosions.

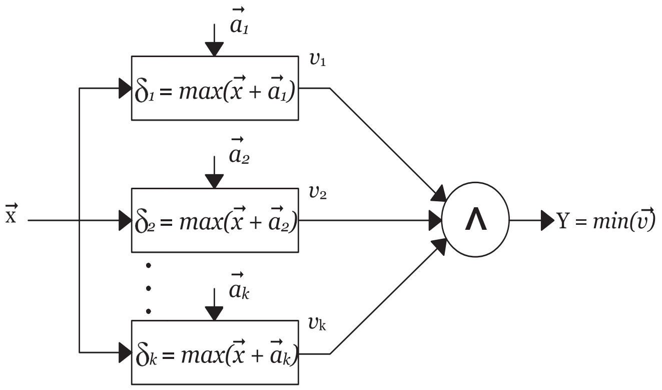

Fig. 2. MMNN architecture used for the Matheron decomposition via dilations.

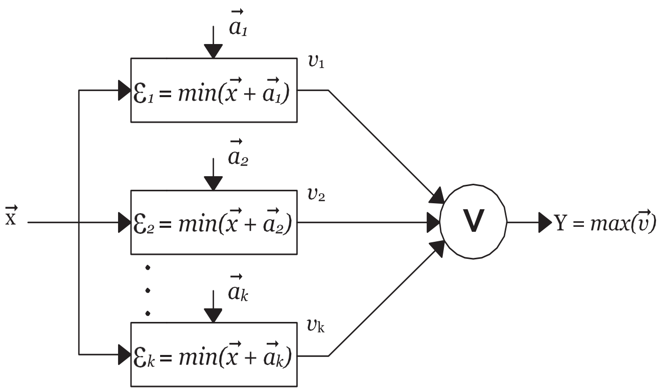

Fig. 3. MMNN architecture used for the Matheron decomposition via erosions.

B. MMNN Architecture for the Banon and Barrera Decomposition

Sousa [1], [2] defined the MMNN for designing translation invariant operators satisfying the Banon and Barrera decomposition theorem [7] for sup-generators as well as for inf-generators. The Banon and Barrera theorem [7] states that every translation invariant operator, not necessarily increasing, may be decomposed by a union of sup-generators or a intersection of inf-generators. Figure 4 presents the architecture of the MMNN for the Banon and Barrera decomposition [7] via sup-generators. The following equations define the MMNN architecture for the Banon and Barrera decomposition [7] via sup-generators according to this approach.

$$u_{k1} = \epsilon_k = min\Big(\vec{x} - \vec{a}_k\Big), \tag{10}$$

$$u_{k2} = \delta_k^a = 1 - max\left(\vec{x} + \vec{b}_k\right). \tag{11}$$

$$\text{Sup-Generator: } v_k = min\left(\vec{u_k}\right), \quad k = 1, 2, \ldots, N, \tag{12}$$

where

$$\vec{u_k} = (u_{k1}, u_{k2}), \quad k = 1, 2, \ldots, N. \tag{13}$$

$$\text{Network output: } Y = max(\vec{v}), \tag{14}$$

where

$$\vec{v} = (v_1, \ v_2, \ \ldots \ , v_N). \tag{15}$$

The weight matrices, A and B, of the MMNN are defined by

$$A = (\vec{a}_1; \ \vec{a}_2; \ \ldots; \ \vec{a}_N), \tag{16}$$

$$B = (\vec{b}_1; \ \vec{b}_2; \ \ldots; \ \vec{b}_N), \tag{17}$$

where $\vec{a}_k$ and $\vec{b}_k \in \mathcal{R}^k$, $k = 1, 2, \ldots, N$, represent the MMNN weights (i.e. rows composed by structuring elements $\vec{a}_k$ and $\vec{b}_k$). Symbol $\wedge$ represents the minimum operation in the sub-integrators units and symbol $\vee$ represents the maximum operation in the general integrator unit.

In a dual manner, the architecture for the Banon and Barrera decomposition [7] via inf-generators is defined by substituting dilations by erosions, anti-dilations by anti-erosions, symbol $\wedge$ by $\vee$ in the sub-integrators units, and symbol $\vee$ by $\wedge$ in the general integrator unit. Figure 5 presents the MMNN architecture for the Banon and Barrera decomposition [7] via inf-generators.

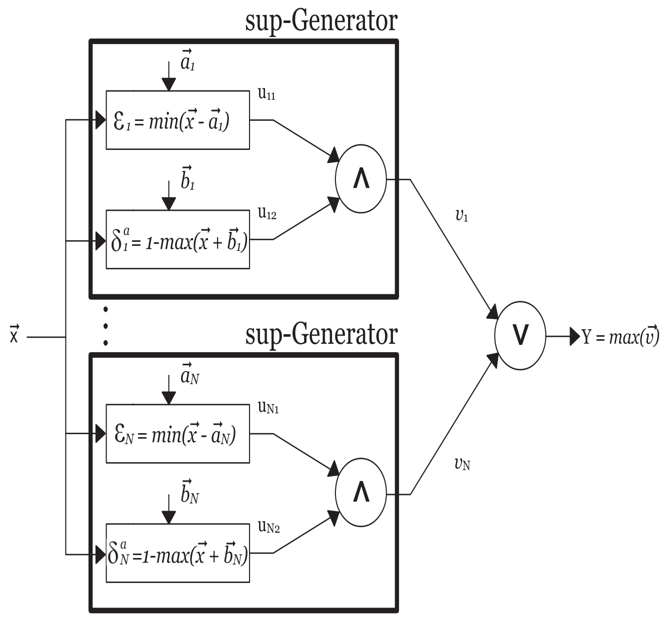

Fig. 4. MMNN architecture used for the Banon and Barrera decomposition via sup-generators.

IV. MMNN Training Algorithm

A. MMNN Training for the Matheron Decomposition

Based on the BP algorithm and the MMNN architecture ilustrated in Figure 2, Sousa [1], [2] presented the MMNN training for the Matheron decomposition [3] as follows:

$$A(n+1) = A(n) - \mu \nabla_A J(A), \; n = 0, 1, \ldots \tag{18}$$

where μ is the learning rate and $\nabla_A J(A)$ is the gradient matrix for some objective function $J(A)$ (to be minimized with respect to the weight matrix A). For a given training set

$$\Big\{(\vec{x}_m, d_m), \; m = 1, 2, \ldots, M\Big\}, \tag{19}$$

where d_m is the desired output for a given input $\vec{x}_m$, $J(A)$ is defined by

$$J(A) = \frac{1}{2} e_m^2, \tag{20}$$

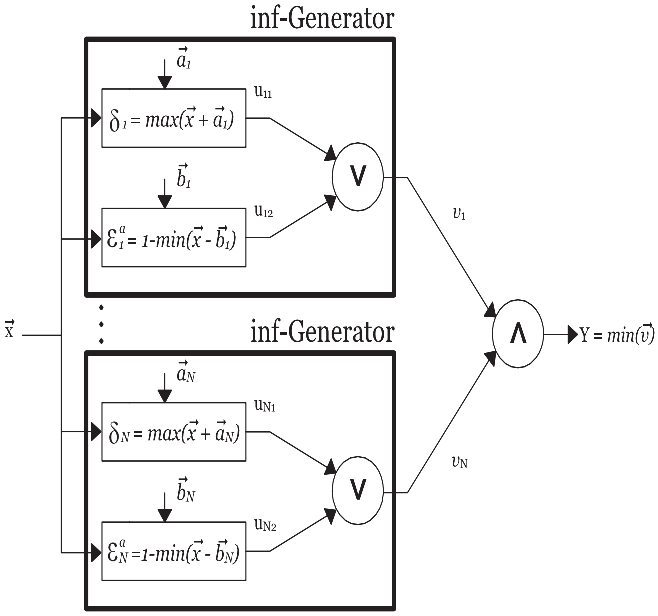

Fig. 5. MMNN architecture used for the Banon and Barrera decomposition via inf-generators.

where $e_m = d_m - y_m$ is the difference between the desired output and the actual output for the input $\vec{x}_m$, $m = 1, 2, \ldots, M$. The gradient presented in equation (18) is given by

$$\frac{\partial J}{\partial \vec{a}_k} = -e \frac{\partial y}{\partial v_k} \frac{\partial v_k}{\partial \vec{a}_k}, \; k = 1, 2, \ldots, N. \tag{21}$$

According to Sousa [1], [2], the partial derivatives in equation (21) are estimated by the methodology of Pessoa and Maragos [19] via rank indication vectors $\vec{c}$ and smooth impulse functions Q_σ, and are given by

$$\nabla_A J(A) = -e.\text{diag}(\vec{c}).C, \tag{22}$$

where

$\vec{c} = \frac{Q_\sigma(y.\vec{1} - \vec{v})}{Q_\sigma(y.\vec{1} - \vec{v}).\vec{1}^T}$ and $C = (\vec{c}_1; \vec{c}_2; \ldots; \vec{c}_N)$, with $\vec{c_k} = \frac{Q_\sigma(v_k.\vec{1} - \vec{x} + \vec{a_k})}{Q_\sigma(v_k.\vec{1} - \vec{x} + \vec{a_k}).\vec{1}^T}$.

B. MMNN Training for the Banon and Barrera Decomposition

Based on the BP algorithm and the MMNN architecture ilustrated in Figure 4, Sousa [1], [2] presented the MMNN training for the Banon and Barrera decomposition [7] as follows:

$$A(n+1) = A(n) - \mu \nabla_A J(A, B), \; n = 0, 1, \ldots \tag{23}$$

$$B(n+1) = B(n) - \mu \nabla_B J(A, B), \; n = 0, 1, \ldots \tag{24}$$

where μ is the learning rate and $\nabla_A J(A, B)$ and $\nabla_B J(A, B)$ are the gradient matrices for some objective function $J(A, B)$ (to be minimized with respect to the weight matrices A and B). For a given training set defined in Equation 19, $J(A, B)$ is defined in a similar fashion by

$$J(A, B) = \frac{1}{2} e_m^2. \tag{25}$$

The gradients presented in equations (23) and (24) are given by

$$\frac{\partial J}{\partial \vec{a}_k} = -e \frac{\partial y}{\partial v_k} \frac{\partial v_k}{\partial u_{k1}} \frac{\partial u_{k1}}{\partial \vec{a}_k}, \; k = 1, 2, \ldots, N, \tag{26}$$

$$\frac{\partial J}{\partial \vec{b}_k} = -e \frac{\partial y}{\partial v_k} \frac{\partial v_k}{\partial u_{k2}} \frac{\partial u_{k2}}{\partial \vec{b}_k}, \; k = 1, 2, \ldots, N. \tag{27}$$

According to Sousa [1], [2], the partial derivatives in equations (26) and (27) are estimated by the methodology of Pessoa and Maragos [19] via rank indication vectors $\vec{c}$ and smooth impulse functions Q_σ, and are given by

$$\nabla_A J(A, B) = -e.\text{diag}(\vec{c}).\text{diag}(\widetilde{c}_1).C_1, \tag{28}$$

$$\nabla_B J(A, B) = -e.\text{diag}(\vec{c}).\text{diag}(\widetilde{c}_2).C_2, \tag{29}$$

where

$$\widetilde{c}_1 = (\widehat{c}_{11}, \widehat{c}_{21}, \ldots, \widehat{c}_{N1}), \; \widetilde{c}_2 = (\widehat{c}_{12}, \widehat{c}_{22}, \ldots, \widehat{c}_{N2}),$$
$$(\widehat{c}_{k1}, \widehat{c}_{k2}) = \frac{Q_\sigma(min(\vec{u}_k).\vec{1} - \vec{u_k})}{Q_\sigma(min(\vec{u}_k).\vec{1} - \vec{u_k}).\vec{1}^T},$$
$$C_1 = (\vec{c}_{11}; \vec{c}_{21}; \ldots; \vec{c}_{N1}), \; \vec{c_{k1}} = -\frac{Q_\sigma(u_{k1}.\vec{1} - \vec{x} + \vec{a_k})}{Q_\sigma(u_{k1}.\vec{1} - \vec{x} + \vec{a_k}).\vec{1}^T},$$
$$C_2 = (\vec{c}_{12}; \vec{c}_{22}; \ldots; \vec{c}_{N2}), \; \vec{c_{k2}} = -\frac{Q_\sigma((1-u_{k2}).\vec{1} - \vec{x} - \vec{b_k})}{Q_\sigma((1-u_{k2}).\vec{1} - \vec{x} - \vec{b_k}).\vec{1}^T}.$$

V. The Proposed Method

The method proposed in this paper, referred to as adaptive hybrid morphological operators design (AHMOD) method, uses an adaptive evolutionary search mechanism for designing translation invariant operators via the Matheron decomposition [3] and the Banon and Barrera decomposition [7] for image restoration of images corrupted by salt and pepper noise. It consists of a hybrid model composed of an MMNN [1], [2] and an adaptive GA [18], which searches for the initial weights, architecture and number of modules of the MMNN, wherein each element of the adaptive GA population is trained via the BP algorithm. Each element of adaptive GA population represents an MMNN. As an example, Figure 6 represents an element of the adaptive GA population, where $se_{(i)}$, $i = 1, 2, \ldots, N$, denotes the initial MMNN weights. The term $arch$ indicates the MMNN architecture (each architecture has a corresponding training algorithm). The term mod represents the number of MMNN modules. Table I presents an example of coding used for identifying the MMNN architectures. Figure 7 illustrates the general scheme of the proposed method.

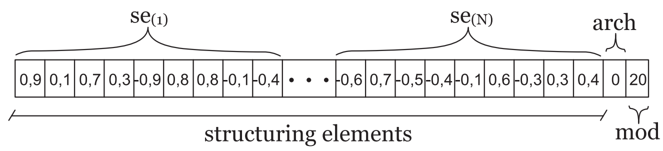

Fig. 6. Coding of the adaptive GA chromosome.

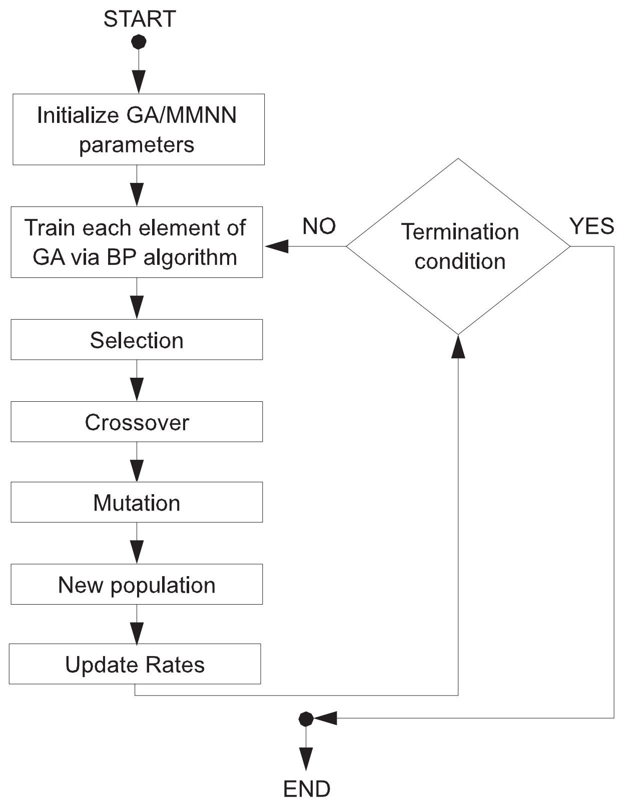

Fig. 7. General scheme of the proposed method.

TABLE I

EXAMPLE OF CODE FOR THE MMNN ARCHITECTURES.

Code	MMNN architecture
0	Matheron decomposition via dilations
1	Matheron decomposition via erosions
2	Banon and Barrera decomposition via sup-generators
3	Banon and Barrera decomposition via inf-generators

VI. SIMULATIONS AND RESULTS

The noise to signal ratio (NSR) is used for assessing performance of the designed operator when applied to restoration of images. It is defined by

$$NSR = 10\, log_{10} \frac{\overline{(D-Y)^2}}{\overline{(D)^2}}, \qquad (30)$$

where $\overline{(D-Y)^2}$ and $\overline{(D)^2}$ represent the mean energy of the error (second moment of the error) and the mean energy of the desired output (second moment of the target).

The MMNN is trained via an adaptive GA with initial population of 100 elements, maximum of 100 generations, with an interval of adaptive variation $Tx_i = 1.0$ to $Tx_f = 0.5$ for crossover probability and $Tx_i = 0.05$ to $Tx_f = 0.0001$ for mutation probability, according to [18]. The adaptive GA stopping criterion is the number of iterations of the GA. Each element of the adaptive GA population is then trained via the BP algorithm for 100 epochs, using smooth rank function ($Q_\sigma = exp\left[-\frac{1}{2}(x/\sigma)^2\right]$) with the smoothing parameter $\sigma = 0.05$, and a convergence factor $\mu = 0.01$. The training set consists of 25% of a given image (that represents a continuous region of the image), whereas the test set consists the entire image. All images are normalized within the range [0,1]; gray-scale structuring elements are normalized in the range [-1,1].

A. Applications

A classical problem in image processing is restoration of images corrupted by noise [20]. The present paper considers salt and pepper noise. The classical median filter [20] is an alternative commonly used for restoring images corrupted by that noise.

The Table II presents the results obtained by the proposed method (AHMOD Method) and by the methods proposed in literature, where MEDIAN denotes the median filter [20] result, MMNN (SGA) and MMNN (BP) denote the results obtained by Sousa [1], [2] and MMNN (AGA) and MMNN (AGA-MOD) denote the results obtained by Araújo et al. [17].

TABLE II

NSR RESULTS FOR A NOISE DENSITY = 5%.

Method	MMNN mod	MMNN arch	NSR (dB)
MEDIAN	-	-	-21,55
MMNN (SGA)	8	1	-17,63
MMNN (SGA)	8	0	-19,00
MMNN (SGA)	25	1	-18,40
MMNN (SGA)	25	0	-19,70
MMNN (BP)	8	1	-19,65
MMNN (BP)	8	0	-21,10
MMNN (BP)	25	1	-22,19
MMNN (BP)	25	0	-23,53
MMNN (AGA)	8	1	-22,21
MMNN (AGA)	8	0	-23,22
MMNN (AGA)	25	1	-23,11
MMNN (AGA)	25	0	-23,76
MMNN (AGA)	50	1	-23,27
MMNN (AGA)	50	0	-24,07
MMNN (AGA)	75	1	-23,31
MMNN (AGA)	75	0	-24,13
MMNN (AGA)	100	1	-24,11
MMNN (AGA)	100	0	-24,27
MMNN (AGA-MOD)	22	0	-24.52
AHMOD Method	24	0	**-24.81**

The AHMOD method automatically chose the MMNN architecture via Matheron Decomposition by dilations ($arch = 0$) and selected the MMNN modules amount ($mod = 24$). It is observed that for a noisy density corresponding to 5%, the proposed method presented better performance when compared to other methods presented in the Table II. It is worth to mention that the AHMOD method overperforms the MMNN (AGA) by using a smaller number of MMNN modules. The AHMOD method led to NSR=-24.81dB with 24 modules, while MMNN (AGA) led to NSR=-24.27dB with 100 modules. The MMNN modules amount decrease is a positive factor when it considers a inquiry of the computational complexity for the morphological operator design. Furthermore, when compared to MMNN (AGA-MOD), the proposed method slightly increased the number of MMNN modules (an increase of two MMNN modules). However, even with a slightly increase in MMNN modules, the proposed method obtained a gain of 0.29dB. Figure 8 shows the image corresponding to the morphological operator obtained by AHMOD method for Matheron decomposition by 24 dilations.

a) Test image

b) Restored image

Fig. 8. Testing images. Results of AHMOD method for Matheron decomposition by 24 dilations. (a) Test image and (b) Restored image.

VII. Conclusions

An adaptive hybrid morphological method for designing translation invariant operators, via Matheron Decomposition and via Banon and Barrera Decomposition, was presented in this paper. It consists of a hybrid model composed of a Modular Morphological Neural Network (MMNN) and an evolutionary search mechanism: the adaptive Genetic Algorithm (GA) (adaptive rates in genetic operators). The proposed method searches for initial weights, architecture and number of modules in the MMNN; then each element of the adaptive GA population is trained via the Back Propagation (BP) algorithm.

Results have shown that the proposed method is more efficient than the median filter and the methods proposed by Sousa [1], [2] and Araújo et al. [17] for restoring images corrupted by salt and pepper noise. The proposed method has reduced the number of decompositions and the computational complexity in the operators design when compared to MMNN (AGA) [17]. Furthermore, when compared to MMNN (AGA-MOD) [17], the proposed method slightly increased the number of MMNN modules. However, even with a slightly increase in MMNN modules, the proposed method obtained a meaningful gain in terms of noise to signal ratio (NSR).

Future works will consider the proposed method in image segmentation and pattern recognition.

Referências

[1] R. P. Sousa. *Design of translation invariant operators via neural network training*. PhD thesis, UFPB, Campina Grande, Brazil, 2000.

[2] R. P. Sousa, J. M. Carvalho, F. M. Assis, and L. F. C. Pessoa. Designing translation invariant operations via neural network training. In *Proc. of the IEEE Intl Conference on Image Processing*, Vancouver, Canada, 2000.

[3] G. Matheron. *Random Sets and Integral Geometry*. Wiley, New York, 1975.

[4] J. Serra. *Image analysis and mathematical morphology*. Academic Press, London, 1982.

[5] S. R. Sternberg. Cellular computers and biomedical image processing. In J. Sklansky and J. C. Bisconte, editors, *Lecture Notes in Medical Iformation, Proc. on Biomedical Images and Computers*, volume 17, pages 274–319. Springer- Verlag, Berlin, 1980.

[6] Henk J. A. Heijmans. Mathematical morphology: A modern approach in image processing based on agebra and geometry. *SIAM Review*, 37(1):1–36, March 1995.

[7] G. J. F. Banon and J. Barrera. Minimal representation for translation invariant set mappings by mathematical morphology. *SIAM J. Appl. Math.*, 51(6):1782–1798, 1991.

[8] P. Yang and P. Maragos. Character recognition using min-max classifiers designed using an LMS algorithm. *Visual Communications and Image Processing*, 92(1818):674–685, nov. 1992.

[9] E. R. Dougherty and R. P. Loce. Efficient design strategies for the optimal binary digital morphological filter: Probabilities, constraints, and structuring-element libraries. In Edward R. Dougherty, editor, *Mathematical Morphology in Image Processing*, pages 43–92. Marcel Decker, Inc., New York, 1993.

[10] J. L. Davidson and F. Hummer. Morphology neural networks: An introduction with applications. *Circuits, System and Signal Process*, 12(2):179–210, 1993.

[11] S. Haykin. *Neural networks: A comprehensive foundation*. Prentice Hall, New Jersey, 1998.

[12] G. X. Ritter. Recent developtments in image algebra. In *Advances in Electronics and Eletron Physics*, volume 80. Academic Press, 1991.

[13] C. B. Herwing and R. J. Shalkoff. Morphological image processing using artificial neural networks. In C. T. Leondes, editor, *Control and Dynamic Systems, Vol. 67*, pages 319–379. Academic Press, 1994.

[14] L. F. C. Pessoa and P. Maragos. Neural networks with hybrid morphological/rank/linear nodes: A unifying framework with applications to handwritten character recognition. *Pattern Recognition*, 33:945–960, 2000.

[15] N. R. Harvey and S. Marshall. The use of genetic algorithms in morphological filter design. *Signal Processing Image Communication*, 8:55–71, 1996.

[16] J. R. F. Oliveira. *Using Genetic Algorithms in Morphological Decomposition of translation Invariant Operators applied to Digital Images*. PhD thesis, INPE, 1998.

[17] R. A. Araújo, F. Madeiro, R. P. Sousa, and L. F. C. Pessoa. Modular morphological neural network training via adaptive genetic algorithm for designing translation invariant operators. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Toulouse, France, 2006.

[18] M. Gen and R. Cheng. *Genetic algorithms and engineering optimization*. John Wiley, New York, 2000.

[19] L. F. C. Pessoa and P. Maragos. MRL-filters: A general class of nonlinear systems and their optimal design for image processing. *IEEE Transactions on Image Processing*, 7:966–978, 1998.

[20] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Publishing Company, New York, 1992.