REDUCED COMPUTATION GENETIC ALGORITHM FOR NOISE REMOVAL

C. Vertan[*], C. I. Vertan[**], V. Buzuloiu[*]

[*] Bucharest "Politehnica" University, Romania
[**] University of Bucharest, Romania

INTRODUCTION

In the last decades, the field of image processing became more and more attractive, sustained by the continuos advance of electrical and computer engineering. The increasing of the computing (processing) power allows to consider successfully various applications. The typical image processing system consists from several different building blocks that performs distinctive tasks. Among them, the noise removal is one of the most common encountered processing steps.

In the same time period, the information engineering introduced new concepts and techniques, and among them the evolutionary computing emerged as a promising solution for several problems. The evolutionary computing domain consists of a set of nature-inspired algorithms and paradigms, as the neural networks and the genetic algorithms.

Many researchers investigated the applications of genetic algorithms in image (or signal) processing, highlighting the benefits obtained by their use in increasing some quality measures or an adaptive behaviour of different classical methods.

This contribution investigates the use of genetic algorithms in image denoising by non-linear (ordering-based) filters. The paper proposes a computational effective method for the genetic-based filter synthesis, that uses a model-free approach (no assumptions being made on the noise distribution or the image contents). The main originality is that the proposed method uses artificially (synthesised) test images for the noise estimation and filter design.

The following outlines the plan for the main part of this contribution; the next two sections (section 2 - "Image Filtering Approaches" and respectively section 3 - "Genetic Algorithms - an Overview") contain the statement of the noise removal problem and the basic problems related to the genetic algorithm's definition. Section 4 (entitled "Filter Synthesis By Genetic Algorithms: Classical Approaches And New Methods") describes the proposed method. Finally, section 5 contains some closing remarks and conclusions, and section 6 lists the references used in the paper.

IMAGE FILTERING APPROACHES

The image processing may be described as a set of "Image In, Image Out" operations (for making a difference with the image analysis, described as "Image In, Description Out"). The area of image processing operations mainly refers to the class of enhancement, restoration and filtering operations, as described in Jain (3). Although there are no exact boundaries between the image enhancement and the filtering, the common interpretation considers the filtering operation as responsible for the noise removal.

Obviously, the image denoising must take into account the noise distribution. Although it appears a simple task, the filter design becomes more complicated when dealing with unknown noises or noise mixtures. The ultimate filter must behave constantly well for any noise type.

The linear filtering assumes a spatial frequency characterization of the noise and is based on two dimensional unitary transforms (Fourier, Cosine, etc.) (3). Although simple to implement, this type of denoising seems to have lost his appeal.

The nonlinear filtering involves local nonlinear operations. The main class of filters is the family of rank order based filters, described in Pitas and Venetsanopoulos (6). The basic principle is to order the pixel values within a moving window across the image; the order statistics are then used genuine (the rank order filters) or as linear combinations (the L-filters) (6).

For a given N-point filtering window (usually square), if the selected pixel values for the current position are denoted by x_1, x_2, ..., x_N, their corresponding order statistics are $x_{(1)} \le x_{(2)} \le \ldots \le x_{(N)}$ and the output of the L-filter for the given position is:

$$y = \sum_{i=1}^{N} a_i x_{(i)} \qquad (1)$$

The filter weights a_i are positive constants, chosen such that their sum is unitary (the normalisation condition).

$$\sum_{i=1}^{N} a_i = 1 \qquad (2)$$

The filter design optimises a quality criterion; typically this criterion is the MSE (Mean Square Error), as defined by equation (3). However, other criteria, such as MAE (Mean Absolute Error) (defined in equation (4)) are also useful in the quality measurement.

$$MSE = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} (f(m,n) - g(m,n))^2 \qquad (3)$$

$$MAE = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} |f(m,n) - g(m,n)| \qquad (4)$$

In the equations above M and N are the image dimensions in pixels, $f(m,n)$ is the value of the original pixel at location (m,n) and $g(m,n)$ is the value of the measured pixel at the same location.

Different types of filters are obtained by changing the weights; if the weights continuously adjust according to the changing image (or noise) statistics, the filter is said to be adaptive. If the noise distribution is known and the noise process is stationary (its statistical parameters are constant along the image) the filter weights can be uniquely determined as constants.

If the noise is non-stationary, or if the noise distribution is unknown, some adaptive algorithms may be used for the computing of weights - such as the LMS-class algorithms, that use some local statistics for the estimation of noise distribution and produce adaptive filters.

GENETIC ALGORITHMS - AN OVERVIEW

The history of genetic algorithms is obviously preceded by the history of modern genetics. Starting with the framework of Charles Darwin, genetics is structured as a real science, and the researches of Gregor Mendel have defined the basic concepts of heritage laws and discrete structure of genetic information. The biology principles and the idea of evolution are now retrieved into algorithmic tools and melted with mathematical concepts (dynamic programming, stability theory). The result of this integration work, led by Holland, is the classical genetic algorithm and its many "clones", as described in Michailewicz (4).

A genetic algorithm instance is defined by the following features:

- *Population*: A finite set of possible solutions (candidate solutions) of the problem, evaluated in the same time.
- *Generative function* : The mechanism of producing new, improved potential solutions from the current population; this function can be split into a selection function and a production function. The selection function chooses the best candidate from the current population, according to the fit of each individual; the production function creates new candidates by combining the selected ones.
- *Reduction function* : The mean for the rejection of the ill-fitted candidates from the population.
- *Stopping criterion (or stop function)*: The checking of some conditions, regarding either the population (the optimality of the individuals), either some external variables (the number of iterations, the elapsed time, etc.)

The selection, production and rejection functions are stochastic, depending on particular realisation of some independent stochastic processes.

Each individual of the population is formed by basic constitutive elements called genes; in the case of classical genetic algorithms (that we will consequently use) the genes are binary symbols. The fit function is problem specific and quantitatively describes the adaptation of an individual for the specified optimisation task. The individuals are ranked according to their fit and the selection function chooses the best fitted specimens to form the new generation. There exist several selection schemes, based on different decision rules about the surviving of an individual; the most used are the roulette-wheel selection and the elitist model (4).

The production function assures the population growth and evolution by two distinctive methods: the mutation and the mating. The mutation function randomly affects the genes of the individuals, that means that the corresponding values toggle. This mechanism is a diversity producer in the next generation; however, the changes are seldom (the usual mutation rate does not exceed 1% from the total number of genes in the population). The mating function produces the so called offspring's: two individuals of the populations produce two offspring's. The usual mechanism is the cross-over: the parents switch parts of their genetic code. Depending on the model, the children's replace the parents, or compete with them.

The reduction function eliminates the ill fitted individuals from the new population, so that the population size remains fixed.

The process of selecting, mating, mutating and reducing the individuals continues, iteratively, until the stopping criterion is reached. The best fitted individual is then the (near) optimal solution.

FILTER SYNTHESIS BY GENETIC ALGORITHMS: CLASSICAL APPROACHES AND NEW METHODS

As shown in the previous section, any genetic algorithm can be viewed as a combinatorial optimisation solver. The use of such a technique for a specific problem is

obvious - it is sufficient to express the problem in terms of optimality, thing that is not so hard (quoting (4) "Whatever we do, we optimise something").

The design of an optimal filter is equivalent to the estimation of a proper set of filter parameters such that the corresponding filter minimises an error criterion. The set of parameters depends strongly on the filter type. Chu (1) uses weights for the design of stack filters; Ostrowski (5) uses the synaptic weights to characterise a neural filter; Harvey and Marshall (2) use the filtering window shape and the succession of basic morphological operation for the design of alternated sequential morphological filters.
In all these approaches, the set of parameters, properly encoded, constitute an individual of a population. The population evolves, and a large number of possible filters are evaluated on a test and trial basis during the evolution. At each stage of evolution, the best individuals (the best filters) are allowed to continue their development. When the evolution of the population stops, the best individual produces the (near)-optimal filter with respect to the given error criterion.

In this contribution, we use the same principle for the design of a (near) optimal L-filter. The parameters that characterise such a filter are its weight's a_i . Each of the N weights (for an N-point filtering window with a determined shape) is uniformly quantized with b bits. The desired precision (or tolerance) for the weights determines the number of quantization levels. The quantization onto 256 levels (b=8) yields to a reasonable compromise between the computational speed and the tolerance in the computation of weights (less than 0.4%). Each individual of the population consists of (N-1)*b bits (genes); since the filter weights are linear dependent, the N-th weight results from the others (with the normalisation condition (2)).

The approach that we will consequently use maximises a composite quality measure, that consists from both the specified figures (MSE, MAE). This quality factor is denoted by Q and has the expression specified by equation (4):
$$Q=1/(MSE*MAE) \quad (5)$$
Maximising this function means to minimise the quadratic and/ or the absolute error, in other words, this leads to a filtered image that is very close to the original one.

The evolution of the population of filters requires, at each generation and for every individual (filter) the evaluation of the performance. This evaluation is a time-consuming process; it consists of the filtering of the noise degraded train image and the computation of the quality measure for the filtered result. Two main drawbacks arise from this approach; they derive from the computational amount due to the image size and the limitation of the filter optimality to the image and noise used for training.

The computational time is proportional to the number of generations, the number of individuals that compose the population, the filter complexity and the size of the training image. For usual size's of a few dozens (up to four) individuals and several hundreds' generations (as presented by (2)), the necessary computing time is huge (equivalent to 10^4 filtering operations). The filter types that were designed until now are not complex: the morphological operations used in (2) are simply minimum and maximum operations; the stack filtering used in (1) is equivalent to Boolean logic operations and the neural filter optimised in (5) is implemented by an inner product and a look-up table. In spite of this inherent simplicity of the operations, the total time required for the filter synthesis is prohibitive. All the quoted contributions fail to mention the actually physical computing time necessary for the design of a single filter.

It is obvious that the reduction of the image size contributes to some computational effectiveness. If the population size or the number of generations decreases, the best individual becomes less adapted (a filter far from the optimal one). We obtain the image size reduction by considering either a downsampled version or parts of the initial training image. The downsampling reduces the detail's accuracy and produces artefacts; that is why we propose the approach of constructing a smaller training image from parts of the initial one. The selected parts are representative areas of the considered image and must contain the most typical of its features: constant intensity areas and edge parts (sharp and blurred). Such types of areas can be chosen according to some local statistics (grey scale range and variance, or contrast scaling, as defined in (3)).

Following this approach, the computational time is reduced by up to 1000 times (if the training image is 8 by 8 pixels). Still the issued filter is image-depended and will not provide the same performance on images that are much different from the one used for training.

We propose a solution that is somehow close to the principle of vector quantization approach: the training image is formed by artificial synthesised blocks. These blocks are also constant intensity areas (of different intensity levels) and different types of edges (sharp, blurred, with various orientations). The (near) optimal filter obtained from this image behaves equally well on all the natural images.

The smallest test image is an 8 by 8 pixel image, consisting of two 4 by 4 pixel constant areas (bright and dark) and two kinds of 4 by 4 pixel edge areas: sharp and blurred. Such a test image is shown in Figure 1.

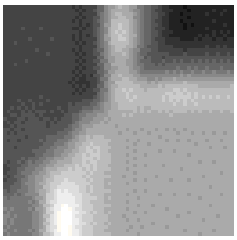

Fig. 1 : Artificial image used for filter weights synthesis

The image shown in Figure 1 corrupted by small Gaussian and impulsive noise, applied on different areas of the image (the noise was no mixture) is the training image for the genetic algorithm. The genetic algorithm was characterised by a population of 20 individuals, 2% mutation and 35% recombination probabilities. The evolution stopped after 500 generations; the best fitted weighting vector obtained is w=(0.0117 0 0.0195 0.2148 0.5547 0.1484 0.0273 0.0039 0.0195).

The result of the filtering by the synthesised L-filter, applied to a mixture noise (Gaussian noise with variance 100 and 10% impulsive noise) is shown in Figure 2. Figure 3 shows the median filtering of the same degraded image, for comparison.

Fig. 2 Filtered image by the proposed L-filter (SNR=19.8 dB, MAE=8.47)

Fig. 3 Median filtered image (SNR=19.8 dB, MAE=8.12)

CONCLUSIONS

The genetic algorithms provide a valuable method for optimisation and have been already used for the synthesis of (near) optimal filters for image noise removal. For the filter synthesis by a genetic algorithm a degraded image and the original image must be available; a large number of possible solutions (filters) are evaluated on a test and trial basis, selecting just the best variants. The approach has two main drawbacks: it requires a huge amount of calculations (which can be solved by a faster machine) and the original image must be available.

This contribution proposes solutions for the announced drawbacks of the method. The computation is reduced by optimising the filter structure for a very small image. The training image is formed by choosing significant parts from the image to be processed; these significant parts will exhibit the highest edge-like and flat region-like characteristics. Still, the knowledge of the non-degraded image is necessary.

We propose a method that does not use the original image. The filter is synthesised on a small (8 by 8 pixels) artificial image. This artificial image contains the representative building blocks of a natural image (flat regions and edges). This test image is degraded by the same (or a similar) noise as the noise affecting the natural image, and the genetic algorithm is used for optimising the 3 by 3 point's window L-filter that obtains the best results in terms of MSE and MAE.

REFERENCES

1. Chu, C. H., 1989, Proc. of ICGA ‘89, 219-224

2. Harvey, N. R., Marshall, S., 1996, Signal Processing: Image Communications, 8, 55-71

3. Jain, A. K., 1989, “Fundamentals of Digital Image Processing”, Prentice Hall Intl., Englewood Cliffs NJ

4. Michailewicz, Z., 1992, “Genetic Algorithms + Data Structures = Evolution Programs”, Springer-Verlag, Berlin

5. Ostrowski, T., Proc. of IEEE Winter Workshop on Nonlinear DSP ‘93, 3.2.1.1-3.2.1.6

6. Pitas, I., Venetsanopoulos, A. N., 1990, “Nonlinear Digital Filters - Principles and Applications”, Kluwer Academic Publ., MA