Multi-level Halftone Image Generation with Genetic Algorithms

Tomoya Umemura, Hernán Aguirre and Kiyoshi Tanaka
Faculty of Engineering, Shinshu University
4-17-1 Wakasato, Nagano, 380-8553 JAPAN

Abstract: - An image halftoning technique that uses a simple GA has proven to be effective generating bi-level halftone images with quality higher than conventional techniques. Many devices are designed to handle more than two halftone levels and a GA based multi-level halftoning technique is desirable. In this paper we extend the bi-level halftoning technique to generate multi-level halftone images. Also we introduce an improved GA (GA-SRM) into the proposed multi-level halftoning technique. Experimental results show that the proposed technique can effectively generate high quality multi-level halftone images and that the inclusion of GA-SRM substantially contributes reducing memory usage and accelerating image generation.

Key-Words: - multi-level image halftoning technique, genetic algorithms

1 Introduction

Recently, Genetic Algorithms[1] (GAs) and their practical application to the solution of problems in various engineering domains are increasingly being investigated[2],[3]. In our work, we focus on the image haltoning technique that is one of the applications of GAs to signal processing.

In [4], a GA is used to generate halftone images with quality higher than conventional techniques such as ordered dithering, error diffusion and so on[5]. It is known that an increase on the number of halftone levels to be displayed contributes to improve the quality of halftone images and many devices are designed to handle more than two levels. The technique mentioned above, however, can generate only bi-level halftone images.

From this point of view, in this paper we extend the bi-level halftone image technique to generate multi-level halftone images to achieve higher quality images. The proposed GA in [4] is a simple GA with an specialized two dimensional crossover. This GA uses a substantial amount of computer memory and processing time. Therefore, we also introduce an improved GA[6],[7] (GA-SRM) into the scheme and try to reduce memory usage and accelerate high quality image generation.

2 Multi-level Halftoning Technique Using GA

2.1 Preparation

In halftoning techniques with GAs the task of the GA is to find the optimum n-gray level pixel pattern corresponding to a given N-gray level input image, where $n < N$. If we try to search for the optimum pixel pattern on the whole image, the chromosome length gets too large and the search space becomes too extensive. For example, for a 256×256 pixels image the chromosome length and search space size would be 65536 and n^{65536}, respectively. A GA is incapable of dealing with these numbers. Therefore, a practical approach is first to divide an input N-gray level image into non-overlapped blocks of $r \times r$ pixels and then use a GA to search the optimum n-level pixel pattern for each image block[4]. Consequently, the GA uses this $r \times r$ two dimensional representation for the chromosome and in the case of bi-level (n=2) halftoning each element of the chromosome $x(i, j) \in \{0, 1\}$. Since we try to generate multi-level halftone images, in this case each element of the chromosome $x(i, j) \in \{0, 1, \cdots, n\text{-}1\}$. **Fig.1** illustrates the block division of an input image and an example of chromosome's representation for n=3.

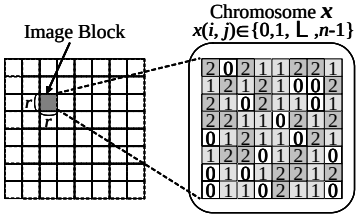

Fig.1 Input image block division and chromosome's representation (n=3, r=8)

2.2 Evaluation

In order to obtain visually high quality halftone images, we evaluate chromosomes with two kinds of evaluation criteria. (i) One is high gray level resolution (local mean gray levels close to the original image), and (ii) the other is high spatial resolution (appropriate contrast near edges)[4]. The bi-level image halftoning

technique calculates a gray level resolution error by

$$E_m = \sum_{(i,j)\in block} \frac{1}{r^2}|g(i,j) - \hat{g}(i,j)| \quad (1)$$

where $g(i,j)$ $(i,j$=0, 1 , $\cdots$, r-1) is the gray level of the (i,j)-th pixel in the input image block, and $\hat{g}(i,j)$ is the estimated gray level associated to the (i,j)-th pixel of the generated halftone block $(x(i,j))$. To obtain $\hat{g}(i,j)$ a reference region around $x(i,j)$ is convoluted by a gaussian filter that models the correlation among pixels. Similar to the bi-level image halftoning technique, **Eq.(1)** is used to generate multi-level halftone images. As mentioned before, however, in this case $x(i,j) \in \{0, 1, \cdots, n\text{-}1\}$ to support the multi-level representation.

In order to preserve the edge information of the input image well, the spatial resolution error in the bi-level image halftoning technique is calculated by

$$E_c = \sum_{(i,j)\in block} \frac{1}{r^2}|G(i,j) - B(i,j)| \quad (2)$$

$$G(i,j) = g(i,j) - \bar{g}(i,j)$$

$$B(i,j) = (x(i,j) - \frac{1}{2})N$$

where $G(i,j)$ is the difference between the gray level $g(i,j)$ of the (i,j)-th pixel in the input image block and its neighboring local mean value $\bar{g}(i,j)$. A positive $G(i,j)$ marks one possible kind of contrast (i.e. a dark pixel on a lighter background) and a negative $G(i,j)$ the other one (i.e. a light pixel on a darker background). The difference between $G(i,j)$ and $B(i,j)$ expresses whether the pixel $x(i,j)$ of the generated block captures the appropriate background of the corresponding pixel $g(i,j)$ of the input image block.

An input gray level pixel $g(i,j)$ can take any value on the range $[0, N]$. The bi-level halftoning technique interprets this range of values as the sole interval I_1 in which the pixel $g(i,j)$ lies and its extremes as the two gray values $g(i,j)$ could map to. In the case of the multi-level halftoning technique, the number of levels an input pixel could map to increases from 2 to n. Therefore, as shown in **Fig.2**, the range $[0, N]$ is split into n-1 intervals and their extremes become the possible values the $g(i,j)$ could map to. The I_k interval is defined by $\frac{k-1}{n-1}N \leq I_k \leq \frac{k}{n-1}N$, where k=1, 2, $\cdots$, n-1.

The $B(i,j)$ term of **Eq.(2)** is accordingly modified to generate multi-level halftone images as shown by

$$B(i,j) = (x(i,j) - \frac{2k-1}{2})\frac{N}{n-1}. \quad (3)$$

Note that before applying **Eq.(3)** we must first determine the interval I_k the pixel $g(i,j)$ belongs to.

The two errors E_m and E_c are combined into one single objective function as

$$E = \alpha_m E_m + \alpha_c E_c \quad (4)$$

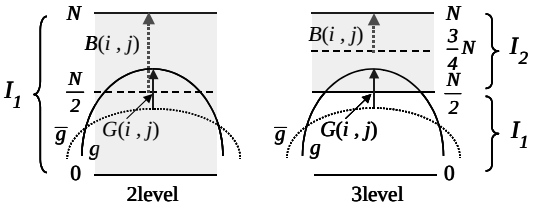

Fig.2 Extension to multi-level halftoning

where α_m and α_c are weighting parameters of E_m and E_c, respectively. The chromosome's fitness is assigned by

$$F = E_{max} - E \quad (5)$$

where E_{max} is the error associated with the worst chromosome in a population. The GA is used to find the optimum compromise between (i) and (ii) with the above fitness function.

3 Improved GA-SRM

The improved GA-SRM for the halftoning problem[6] is based on a model of GA that puts genetic operators in a cooperative-competitive stand with each other[7]. The main features of the model are (i) two genetic operators to create offspring: Self Reproduction with Mutation (SRM) that puts emphasis on mutation, and Crossover and Mutation (CM) that puts emphasis on recombination, (ii) an extinctive selection mechanism, and (iii) an adaptive mutation schedule that varies SRM's mutation rates from high to low values based on SRM's own contribution to the population. The block diagram of the improved GA-SRM is shown in **Fig.3**.

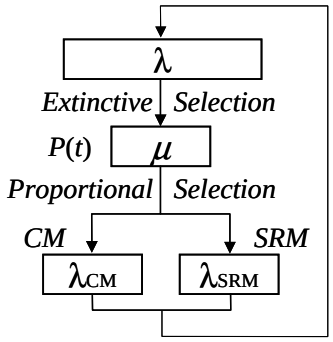

Fig.3 Block diagram of improved GA-SRM

3.1 CM and SRM for Halftoning Problem

In the halftoning problem an individual is represented as a $r \times r$ two dimensional structure. The implementation of CM's and SRM's mutation take into account this two dimensional structure.

CM first crosses over two previously selected parents from $P(t)$ interchanging either their rows or columns at a random chosen position similar to [4]. Then it applies standard bit flipping mutation inverting bits with

a small mutation probability per bit, $p_m^{(CM)}$, analogous to canonical GAs. Thus, mutation in CM is of a quantitative nature after which the number of bits 0s, 1s, $\cdots$, n-1s may change. CM creates λ_{CM} individuals at each generation.

SRM, on the other hand, creates an exact copy of a previously selected parent from $P(t)$ and then applies mutation only to the bits inside a mutation block (square region). SRM creates λ_{SRM} individuals at each generation.

SRM is provided with an Adaptive Dynamic-Block (ADB) mutation schedule. With ADB the mutation block area $\ell \times \ell$ is dynamically adjusted every time a normalized mutants survival ratio falls under a threshold, $\gamma < \tau$, following a decreasing approach as shown in **Fig.4**. The offset position $S=(a, b)$ of the mutation block for each chromosome is chosen at random. The normalized mutant survival ratio is specified by

$$\gamma = \frac{\mu_{SRM}}{\lambda_{SRM}} \cdot \frac{\lambda}{\mu} \tag{6}$$

where μ is the number of individuals in the parent population $P(t)$, μ_{SRM} is the number of individuals created by SRM present in $P(t)$ after selection and λ is the total offspring number, $\lambda_{CM}+\lambda_{SRM}$ (see **Fig.3**).

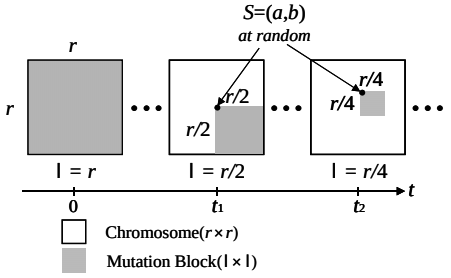

Fig.4 Adaptive Dynamic-Block mutation(ADB)

Two kinds of mutation schemes are investigated for ADB: (i) quantitative and (ii) qualitave mutation. Quantitative mutation in ADB is implemented as the standard bit flipping process. Mutation probability for the bits inside the segment is $p_m^{(SRM)}=\alpha$. After this kind of mutation has been applied, the contrast near edges and the local mean average might change in an individual affecting both E_c and E_m in **Eq.(1)**,**(2)**. Quantitative mutation would allow to observe the general effect of parallel mutation in this problem.

On the other hand, qualitative mutation in ADB is implemented as a bit swapping process. First, a set Z is initialized with every bit in the mutation block. A pair of bits are randomly marked and then swapped. The marked bits are removed from Z and the process is repeated until there are no remaining bits in Z. **Fig.5** enlarges a 4×4 mutation block to illustrate the bit swapping process for two of the bits. Note that it is not necessary to set a mutation probability in qualitative mutation since all pairs of bits within the mutation block are simply swapped.

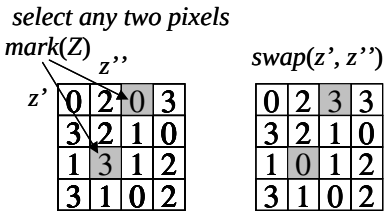

Fig.5 bit swapping process

3.2 Selection

(μ, λ) Proportional Selection[8] implements the required extinctive selection mechanism. Selection probabilities are computed by

$$p(\boldsymbol{x}_i^{(t)}) = \begin{cases} \dfrac{f(\boldsymbol{x}_i^{(t)})}{\sum_{j=1}^{\mu} f(\boldsymbol{x}_j^{(t)})} & (1 \le i \le \mu) \\ 0 & (\mu < i \le \lambda) \end{cases} \tag{7}$$

where $\boldsymbol{x}_i^{(t)}$ is an individual at t-th generation which has the i-th highest fitness value $f(\boldsymbol{x}_i^{(t)})$, μ is the number of parents and λ is the number of offspring. Also, selection is reinforced to assure that the two parents selected for crossover are different avoiding that an individual crosses with itself,[1] i.e. the parents for crossover are $\boldsymbol{x}_i^{(t)}$ and $\boldsymbol{x}_j^{(t)}$ $(i \neq j)$.

The extinctive nature of this selection mechanism subjects SRM's and CM's offspring to compete for survival.

4 Results and Discussion

4.1 Experimental Setup

A simple GA similar to the one proposed in [4] (CM and Proportional Selection) denoted as cGA and the improved GA (CM, SRM, and (μ, λ) Proportional Selection) denoted as GA-SRM are used to generate multilevel halftone images. In the case of the simple GA the crossover probability is set to p_c=1.0 and mutation probability to p_m=0.001. In the case of GA-SRM, mutation probability for CM is set to $p_m^{(CM)}$=0.001, the ratio for offspring creation to $\lambda_{CM} : \lambda_{SRM}$=1 : 1, and the extinctive ratio to $\mu : \lambda$=1 : 2. Also, we use a τ=0.40 as a threshold for the normalized mutant survival ratio. Mutation probability for ADB when it is implemented with quantitative mutation is set to $p_m^{(SRM)}$=0.125. Again, note that it is not necessary to set a mutation probability in qualitative mutation.

To test the algorithms we use SIDBA's benchmark images in our simulation. The size of the original image is 256×256 pixels with N=256 gray levels. An image is divided into 256 blocks, each one of size 16×16 pixels. For each block, the algorithms were set with different seeds for the random initial population.

[1] We do not check whether two individuals have identical genetic information.

Halftone images are generated for n={3, 4, 8, 16} levels with the weighting parameters set to α_m=0.2 and α_c=0.8. Unless indicated otherwise, results presented here are for the experimental image "Lenna" generated with n=3 levels.

4.2 Performance by Simple GA and Improved GA-SRM

We set the cGA with a population size of 200 individuals and run the algorithm until visually high quality images are achieved. The image quality achieved by the cGA and the number of evaluations it expends (see **4.4**) are used as a reference for comparison in our study.

In order to observe the performance by the proposed algorithm GA-SRM, i.e. the evolution of image quality and its convergence speed, we set the population size to μ=λ_{CM}=λ_{SRM}=100. With this values GA-SRM creates the same number of offspring (200 offspring from 100 parents) as cGA does (200 offspring from 200 parents) to attain high qualtiy images. **Fig.6** shows the image's average-error transition, calculated as the average of the best chromosomes' error in the 256 image blocks, by the two schemes. From **Fig.6** it can be seen that GA-SRM converges faster and reaches better quality than cGA. Also, qualitative mutation performs better than quantitative mutation. Under this population configuration, GA-SRM needs only $0.17T$ evaluations to surpass the final image quality obtained by cGA when qualitative mutation is used (GA-SRMs) whereas $0.25T$ evaluations are needed in the case of quantitative mutation (GA-SRMf).

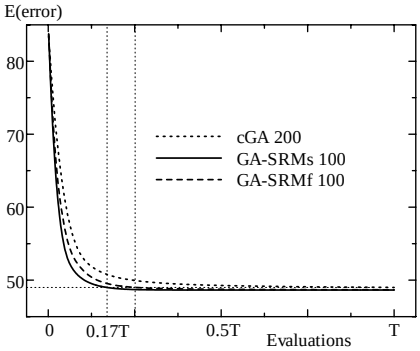

Fig.6 cGA and GA-SRM's performance

SRM's behavior can be observed from **Fig.7**, which presents the block's side length reduction, ℓ, and the number of individuals produced by SRM-ADB that survive selection, μ_{SRM}, for one image block. From this figure it is clear that (i) SRM contributes with beneficial mutations (carried by mutants that survive selection) in every generation of the search process, and (ii) the key factor for SRM to be an effective operator lies in its own regulation mechanism: mutation block adjusted every time the number of mutants that survive selection falls under a minimum level τ.

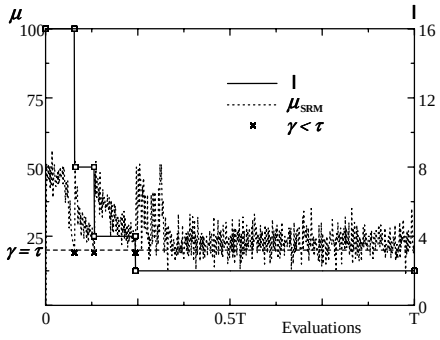

Fig.7 Mutation block's side length reduction and SRM-ADB offspring that survive selection

4.3 Effect of Population Size Reduction

GAs with bigger populations would converge in the long run to lower errors than GAs with smaller populations. From a practical application point of view, however, rather than achieving the minimum possible error we are interested in finding timely and efficiently an upper bound of the minimum error such that i) it guarantees visually high quality images, and ii) further error reductions, although possible, are no longer visually significant. Thus, since reducing memory is an important issue in this application we also observe the performance of the GAs with smaller populations. The error achieved by cGA using a 200 population after T evaluations (cGA 200) satisfies our criteria and is used as the minimum error upper bound in our analysis of population size reduction.

Fig.8 show results by cGA using {200, 100, 20, 4} population configurations. **Fig.9** and **Fig.10** present results for equivalent configurations μ=λ_{CM}= λ_{SRM}={100, 50, 10, 2} by GA-SRMf and GA-SRMs, respectively, along with those obtained by cGA 200.

From **Fig.8** we can see that the cGA with a population of 100 speeds up convergence and can achieve similar image quality than the cGA 200. When the population is further reduced to 20 or 4, however, the algorithm converges to an error higher than cGA 200 and ends with a poor image quality. A population size of 2 individuals was tried for cGA obtaining a final image's average-error above 93.

Fig.9 shows that the introduction of quantitave mutation allows us to considerably reduce population sizes from 100 to 10 and still obtain a gain on search speed to generate images with quality better than cGA 200. However, a further reduction in population sizes from 10 to 2 is not effective. In this minimum configuration the levels of mutation introduce by GA-SRMf are too high, which does not allow SRM's offspring to compete properly against CM's offspring.

In **Fig.10** we observe that GA-SRMs using qualitative mutation with smaller populations converge faster and produce better image quality than the one obtained by cGA 200. In this case, qualitative mutation allows to reduce the population configuration to its minimum level. This is because SRM with this kind of muta-

tion always contributes to introduce diversity in levels such that SRM could be competitive with CM regardless of the population size, avoiding premature convergence.[2] It should be noticed that the probability of cloning with this operator is higher when the mutation block's length has reached its minimum length. In this way qualitative mutation also introduces a kind of implicit elitism. These characteristic explains the GA-SRM's robust performance even with tiny populations and allows us to choose the smallest memory configuration for n=3 to generate halftone images without compromising the image quality. In fact, for "Lenna" the improved GA-SRM using qualitative mutation with μ=2 and λ=4 configuration attained after only $0.08T$ evaluations the same image quality obtained by cGA 200 after T evaluations.

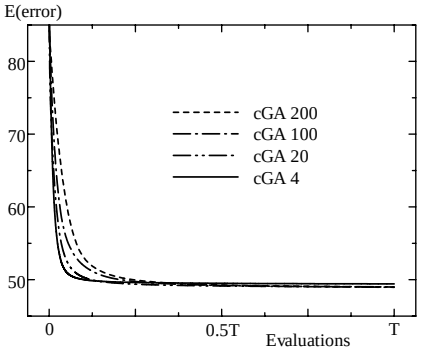

Fig.8 Performance by cGA with different population sizes

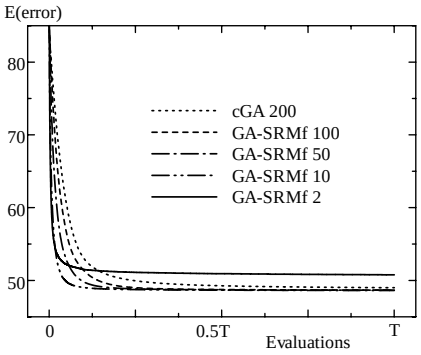

Fig.9 Performance by GA-SRMf with different population sizes

4.4 Effect of Increasing the Number of Halftone Levels (n)

The search space for each image block is $n^{r\times r}$. Thus, keeping constant the image block size ($r \times r$), the sole increase of the number of halftone levels (n) substantially enlarges the size of the search space. A larger search space, in turn, affects the performance of the genetic algorithms.

On the one hand, an increase on n implies an augment on the number of evaluations the algorithms perform to achieve high quality images. For example, from

[2]All the individuals are trapped on local optima in earlier generations.

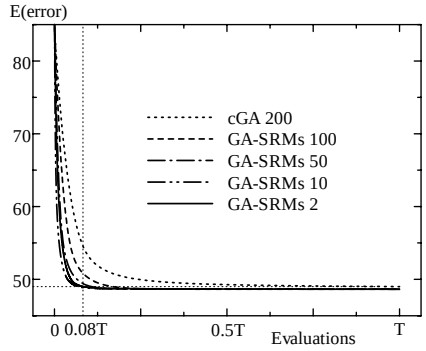

Fig.10 Performance by GA-SRMs with different population sizes

our visual subjective assesment the cGA 200 would require about T={T_3, T_4, T_8, T_{16}}={1.6×10^5, 2×10^5, 4×10^5, 6×10^5} evaluations to generate halftone images with n={3, 4, 8, 16} levels, respectively.

On the other hand, the quality of image achieved by reduced population GAs may also be affected by an increase of n. For example, A GA-SRMs with a minimum memory configuration of μ=2 and λ=4 is effective for n={2, 3, 4, 8}. However, for n={16} GA-SRMs with this memory configuration approaches but can not achieve the same image quality as cGA 200. A cGA with a population of 20 individuals even for n=3 does not reach the image quality of the cGA 200.

As mentioned above, the number of evaluations T used as an stopping criteria are obtained after a subjective assesment of the image quality achieved by cGA 200 and are useful to study the general behavior of cGA and GA-SRM. An objective method for quality assesment should be incorporated within the method to provide precise values of T for this and other images with different characteristics.

After observing the performance of cGA and GA-SRM with various population configurations generating halftone images for n={3, 4, 8, 16} levels we observed that GA-SRM significantly contributes reducing memory usage and accelerating image generation, which are important issues in this application. With regard to memory usage, the cGA can be used with an smaller population configuration. A configuration of 100 parents and 100 offspring for cGA (cGA 100) is enough for any value of n={3, 4, 8, 16}. In the case of GA-SRM, however, a μ=4 parents and λ=8 offspring configuration (GA-SRM 4) would be sufficient. Concerning processing time, for the values of T chosen for cGA 200, **Table 1** illustrates the number of evaluations that GA-SRMs 2 and GA-SRM 4 would require to generate images with quality similar to cGA 200.

Table 1 Evaluations performed by GA-SRMs 2 and 4 to match cGA 200's image quality ("Lenna")

n	3	4	8	16
Search Space	$3^{16\times16}$	$4^{16\times16}$	$8^{16\times16}$	$16^{16\times16}$
cGA 200	T_3	T_4	T_8	T_{16}
GA-SRMs 2	$0.08T_3$	$0.10T_4$	$0.30T_8$	–
GA-SRMs 4	$0.07T_3$	$0.09T_4$	$0.09T_8$	$0.09T$

n=3, 0.07 T_3 n=4, 0.09 T_4 n=8, 0.09 T_8 n=16, 0.09 T_{16}

(a)cGA (200 parent individuals)

n=3, 0.07 T_3 n=4, 0.09 T_4 n=8, 0.09 T_8 n=16, 0.09 T_{16}

(b)GA-SRMs (4 parent individuals)

Fig.11 Halftone images generated by cGA and GA-SRM

Fig.11 shows for each n={3, 4, 8, 16} the images generated by GA-SRMs 4 and cGA 200 after the fraction of evaluations required by GA-SRMs 4 have elapsed. Note that the images generated by GA-SRMs 4 at the respective fraction of T have achieved high quality. On the other hand, the images generated by cGA 200 still require further refinement. It should also be noted that as n increases the difference between images becomes less apparent.

5 Conclusions

In this paper, we have extended a bi-level halftoning technique with GA to a multi-level halftoning technique with GA. Also, an improved GA (GA-SRM) was introduced within the multi-level halftoning technique.

High quality images were generated for n={3, 4, 8, 16} halftone levels. We observed that an increase on the number of levels improves the quality of the halftone image. However, it also has an impact on the performance of the algorithms.

The improved GA-SRM outperforms the simple GA. GA-SRM allows to reduce significantly processing time and memory usage to generate multi-level halftoning images.

As future works, an objective stopping criteria should be designed and introduced in the multi-level halftoning technique. Also, the application to color halftone image and the introduction of multi-objective GA to this scheme should be investigated.

References:

[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, 1989.

[2] K. S. Tang, K. F. Man, S. Kwong and Q. He, "Genetic Algorithms and Their Applications", *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 22-37, Jun. 1996

[3] K. F. Man, K. S. Tang, S. Kwong and W. A. Halong, *Genetic Algorithms for Control and Signal Processing*, Springer-Verlag, 1997

[4] N. Kobayashi and H. Saito, "Halftoning Technique using Genetic Algorithm", *Proc. IEEE ICASSP'94*, pp. 105-108, 1994

[5] R. Ulichney, Digital Halftoning, MIT Press, Cambridge, 1987.

[6] H. Aguirre, K. Tanaka and T. Sugimura, "Accelerated Image Halftoning Technique Using Improved Genetic Algorithm", *IEICE Trans. Fundamentals*, Vol.E83-A, No.8, pp.1566-1574, 2000.

[7] H. Aguirre, K. Tanaka and T. Sugimura, "Cooperative Model for Genetic Operators to Improve GAs", *Proc. IEEE ICIIS'99*, pp. 98-106, Nov. 1999.

[8] T. Bäck and F. Hoffmeister, "Extended Selection Mechanism in Genetic Algorithms", *Proc. 4th Int'l Conf. on Genetic Algorithms*, Morgan Kaufmann, pp. 92-99, 1991.