ELSEVIER

Fuzzy Sets and Systems 103 (1999) 285–293

FUZZY sets and systems

Restoration of gray images based on a genetic algorithm with Laplacian constraint

Yen-Wei Chen [a,*], Zensho Nakao [a], Kouichi Arakaki [a], Xue Fang [a], Shinichi Tamura [b]

[a] *Department of Electrical and Electronic Engineering, Faculty of Engineering, University of the Ryukyus, 1 Senbaru, Nishihara, Okinawa 903-01, Japan*

[b] *Medical School, Osaka University, 2-2 Yamada-oka, Suita, Osaka 565, Japan*

Received May 1998

Abstract

Genetic algorithms are used for restoration of gray images. The restoration problem is modeled as an optimization problem, whose cost function is minimized based on mechanics of natural selection and natural genetics. Because the complicated *a priori* constraints can be easily incorporated by the appropriate modification of the cost function, the algorithm is well suited to the solution of ill-posed problem. The possibility has been demonstrated with computer simulations. A parallel implementation on a multi-workstation environment has been implemented. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Genetic algorithm; Image restoration; Gray level; Laplacian constraint; Parallel implementation

1. Introduction

Many image processing applications, such as satellite remote sensing, medical and scientific imaging, require a high-resolution image. However, currently available optical or imaging system have certain physical limitations, and the image $g(x,y)$ we usually observe is a degraded one by the convolution of the original image $f(x,y)$ and the point spread function (blurring function) $h(x,y)$ of the optical or imaging system, which can be expressed as

$$\begin{aligned} g(x,y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x',y')h(x-x',y-y')\,\mathrm{d}x'\,\mathrm{d}y' \\ &\quad + n(x,y) \\ &= f(x,y) * h(x,y) + n(x,y), \qquad (1) \end{aligned}$$

where $*$ denotes the convolution operator and $n(x,y)$ is the random noise contained in the blurred image. In order to recover the original image $f(x,y)$ from the blurred image $g(x,y)$, we have to deconvolve the $g(x,y)$ with the blurring function $h(x,y)$. The inverse filter [11], whose point spread function is the inverse of the blurring function, and other many methods [1,5,8] have been used for deconvolution problem. If the blurred image is with noise, or if the inverse of $h(x,y)$ does not exist, the image restoration problem becomes an ill-posed problem and it is impossible to recover the original image by the linear technique [11].

The image restoration problem can be viewed as an optimization problem which can be solved by a genetic algorithm (GA). There have been a few researches in application of GAs to image restoration problem. Because the complicated *a priori* constraints

* Corresponding author. Tel.: +81 98 895 87 03; fax: +81 98 895 87 08; e-mail: chen@tec.u-ryukyu.ac.jp.

can be easily incorporated by the appropriate modification of the cost function, the GA method is well suited for solving this ill-posed problem. Takatsu et al. [12] have shown the possibility of the GAs for ill-posed problem with a concept of line process [7]. In our previous work, we have shown the possibility of using GAs for two-dimensional blind deconvolution problem [2]. However, these GA methods are only used for restoration of a binary image. In this paper, we present a GA method for restoration of gray images and propose to use a simple Laplacian constraint for ill-posed problem.

The paper is organized as follows. The basis of the genetic algorithm is given in Section 2 and the constraint used for ill-posed problem is discussed in Section 3. Simulation results are presented in Section 4. Parallel implementation of GA for image restoration is presented in Section 5. Finally, we conclude in Section 6 with a summary and directions for the future research.

2. The genetic algorithm

The typical flowchart of genetic algorithms for image restoration is shown in Fig. 1.

2.1. Image encoding

We use a two-dimensional array as a "chromosome" to represent the image. The size ($N \times N$) of the two-dimensional array is the same as the size of the image and the allele value (0–127) of the chromosome corresponds to the pixel intensity of the image (128 gray level). The illustrative example is shown in Fig. 2. Fig. 2a is an image with 128 gray levels (15×15 pixels), Fig. 2b is its profile (approximation of Gaussian) taken across the center in horizontal directions, and Fig. 2c is its two-dimensional chromosome.

2.2. Initial population and fitness measure

We first create some *pop_size* number of chromosomes (estimates of the original image) randomly in initialization process. And then the blurred image is calculated for each chromosome (estimate) by convolving the chromosome with the blurring function.

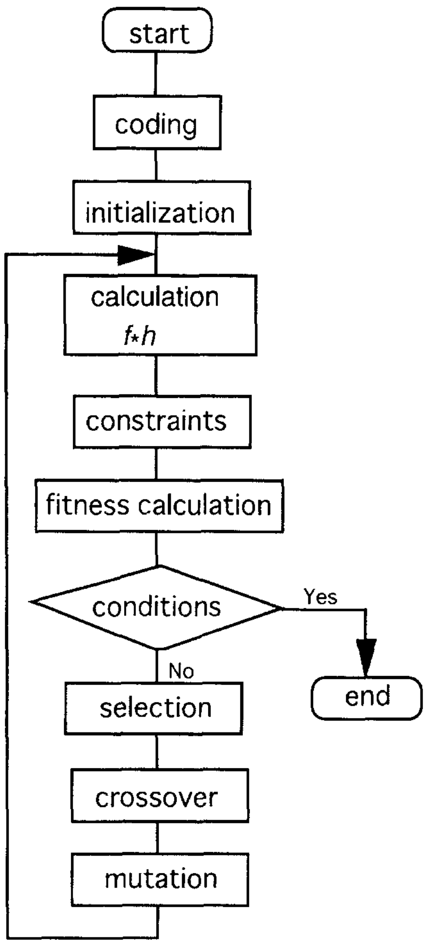


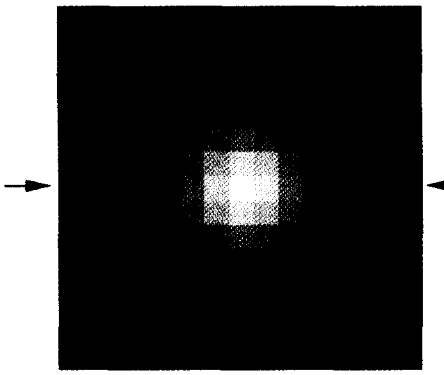
Fig. 1. A typical flowchart of genetic algorithms for image restoration.

The fitness for each chromosome is evaluated by comparing the calculated blurred image of the chromosome with the original blurred image. We want to recover an image f whose blurred image is close to the given image g. The mean-square error between the two blurred images as shown in Eq. (2) is used as the fitness measure or cost function for evaluation.

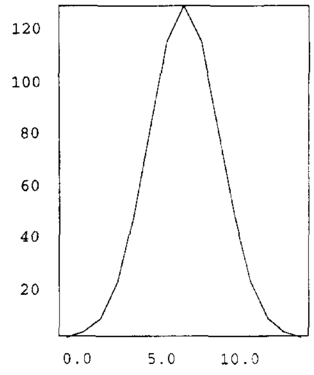
$$E = \|g - \hat{f} * h\|^2, \tag{2}$$

where $\hat{f}$ is the estimate (or chromosome) of f. The lower the cost is, the higher the fitness. The "optimum" solution (restored image) can be obtained by minimizing the cost function of Eq. (2).

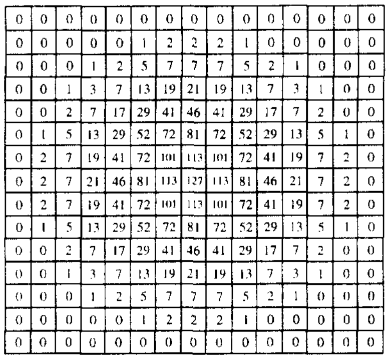
Three genetic operators (selection, crossover and mutation) are performed on the whole population to



(a)



(b)



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	2	2	2	1	0	0	0	0	0
0	0	0	1	2	5	7	7	7	5	2	1	0	0	0
0	0	1	3	7	13	19	21	19	13	7	3	1	0	0
0	0	2	7	17	29	41	46	41	29	17	7	2	0	0
0	1	5	13	29	52	72	81	72	52	29	13	5	1	0
0	2	7	19	41	72	101	113	101	72	41	19	7	2	0
0	2	7	21	46	81	113	127	113	81	46	21	7	2	0
0	2	7	19	41	72	101	113	101	72	41	19	7	2	0
0	1	5	13	29	52	72	81	72	52	29	13	5	1	0
0	0	2	7	17	29	41	46	41	29	17	7	2	0	0
0	0	1	3	7	13	19	21	19	13	7	3	1	0	0
0	0	0	1	2	5	7	7	7	5	2	1	0	0	0
0	0	0	0	0	1	2	2	2	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(c)

Fig. 2. An illustrative example of image encoding. (a) A image with 128 gray levels (15×15 pixels), (b) its profile (Gaussian) taken across the center in horizontal directions, and (c) its two-dimensional chromosome.

guide the chromosomes toward the optimum solution (minimum cost).

2.3. Genetic operators

2.3.1. Selection

We used an elitist selection scheme [10] for the selection process. In this scheme, the best chromosome (estimate) with the lowest cost (the highest fitness) is selected as an elite, which is copied directly into the new population without any change. The other chromosomes (estimates) of the new population are selected by using a roulette selection scheme [10]. In this scheme, a roulette wheel with slot size proportional to fitness values is used. We spin the roulette wheel *pop_size*-1 times; each time we select one chromosome for the new population. Obviously, the best chromosomes get more copies, the average ones stay even, and the worst ones die off. Furthermore, some chromosomes of this new population undergo alterations by means of crossover and mutation.

2.3.2. Crossover

Crossover combines the features of two parent individuals to form two similar offspring by swapping corresponding segments of the parents. The intention of the crossover operator is information exchange between different potential solutions. Four crossover operators are used.

The first one is called *uniform R/C crossover* [2]. Two selected parents exchange their row or column information at the same position. If $\boldsymbol{b}^t = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{pos}, \ldots, \boldsymbol{b}_N)$ and $\boldsymbol{c}^t = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{pos}, \ldots, \boldsymbol{c}_N)$ are to be crossed, the resulting offspring are $\boldsymbol{b}^{t+1} = (\boldsymbol{b}_1, \ldots, \boldsymbol{c}_{pos}, \ldots, \boldsymbol{b}_N)$ and $\boldsymbol{c}^{t+1} = (\boldsymbol{c}_1, \ldots, \boldsymbol{b}_{pos}, \ldots, \boldsymbol{c}_N)$, where the random number *pos* indicates the position of the crossing point and t is the generation number. This operator performs a local small-scale exchange.

The second one is called *random R/C crossover* [2]. The crossing positions for two parents are randomly selected. They can exchange their information at the different position. If $\boldsymbol{b}^t = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{pos1}, \boldsymbol{b}_{pos1+1}, \ldots, \boldsymbol{b}_N)$ and $\boldsymbol{c}^t = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{pos2}, \boldsymbol{c}_{pos2+1}, \ldots, \boldsymbol{c}_N)$ are to be crossed, the resulting offspring are $\boldsymbol{b}^{t+1} = (\boldsymbol{b}_1, \ldots, \boldsymbol{c}_{pos2}, \boldsymbol{b}_{pos1+1}, \ldots, \boldsymbol{b}_N)$ and $\boldsymbol{c}^{t+1} = (\boldsymbol{c}_1, \ldots, \boldsymbol{b}_{pos1}, \boldsymbol{c}_{pos2+1}, \ldots, \boldsymbol{c}_N)$, where the random numbers *pos*1 and *pos*2 indicates the positions of the crossing point for $\boldsymbol{b}$ and $\boldsymbol{c}$, respectively. This operator performs a large-scale exchange.

The third one is called *uniform arithmetic crossover* [10] which is defined as a linear combination of two selected parents. If $\boldsymbol{b}^t$ and $\boldsymbol{c}^t$ are to be crossed, the resulting offspring are $\boldsymbol{b}^{t+1} = a\boldsymbol{b}^t + (1-a)\boldsymbol{c}^t$ and $\boldsymbol{c}^{t+1} = a\boldsymbol{c}^t + (1-a)\boldsymbol{b}^t$, where the parameter a is a constant.

The fourth one is called *non-uniform arithmetic crossover* [10], which is almost same with the previous *uniform arithmetic crossover*. The parameter a is a variable whose value depends on the age of population as follows:

$$a = (1 - t/T) * 0.5, \tag{6}$$

where T is the maximal generation number.

Table 1
Fixed parameters for genetic algorithm

	Object 1	Object 2
Population size	60	50
Probability of crossover 1	0.8	0.3
Probability of crossover 2	0.4	0.9
Probability of crossover 3	0.0	1.0
Probability of crossover 4	0.0	1.0
Probability of mutation 1	0.04	0.15
Probability of mutation 2	0.0	0.3

2.3.3. *Mutation*

Two mutation operators are also used for generation of offspring together with the crossover operators. The intention of the mutation operator is the introduction of some extra variability into the population.

The first one is called *uniform mutation*. It performs a small-scale survey of the area surrounding the pixel selected for mutation and mutates it in the direction that enhances the smoothing of the image. If b_{ij} is the pixel selected for mutation, it is replaced by the averaged value of the surrounding 8 pixels. The second one is called *non-uniform mutation* [10]. If b_{ij} is the pixel selected for mutation, it is replaced by a value b'_{ij}, with

$$b'_{ij} = \begin{cases} b_{ij} + \Delta(t, U - b_{ij}) & \text{if a random digit is 0,} \\ b_{ij} - \Delta(t, L - b_{ij}) & \text{if a random digit is 1,} \end{cases} \tag{7}$$

where U is the upper limit which is 127 and L is the lower limit which is 0. The function $\Delta(t, y)$ [8] is defined as

$$\Delta(t,y) = y\left(1 - r^{(1-(t/T))^b}\right), \tag{8}$$

where T is the maximal generation number, r is a random number from [0, 1], and b is a system parameter determining the degree of non-uniformity. The function $\Delta(t, y)$ returns a value in the range $[0, y]$ such that the probability of $\Delta(t, y)$ being close to 0 as increasing the generation number t. Thus the mutation is responsible for the fine tuning capabilities as increasing the generation number.

3. Constraint

If the degraded image is with noise, the cost function of Eq. (2) must satisfy

$$\|g - \hat{f} * h\|^2 = \varepsilon, \tag{9}$$

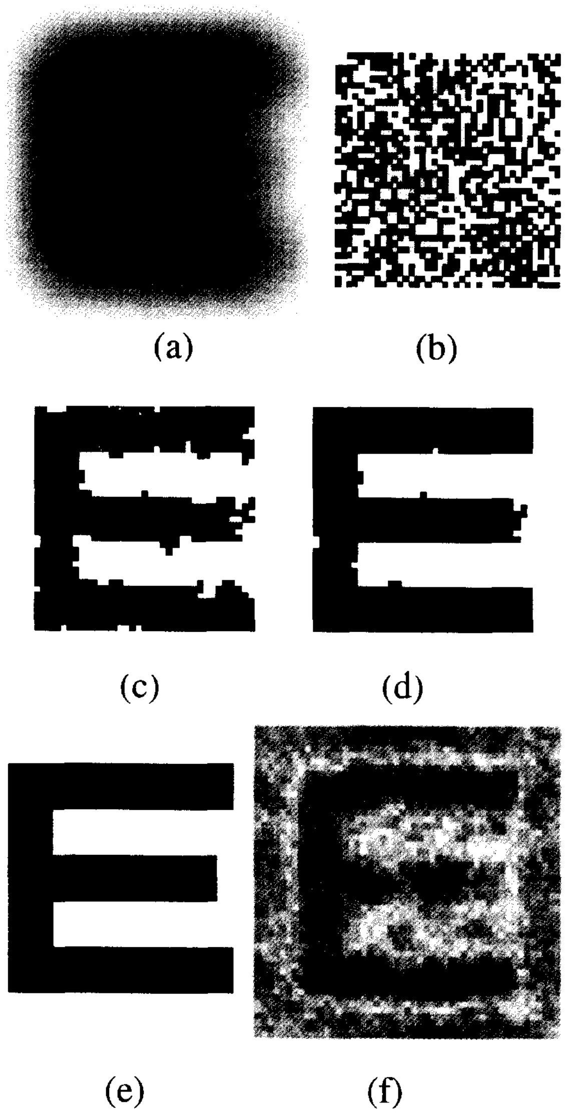


Fig. 3. Genetic restoration of a binary image. (a) Blurred image with an additive Gaussian noise, (b) restored images after 1 generation, (c) restored image after 200th generation, (d) restored after 400th generation, (e) restored after 1000th generation, and (f) restored image by Wiener filter.

where ε is a parameter determined by the variance of noise. Eq. (9) is an ill-posed equation. There may be many solutions f such that Eq. (9) is satisfied. In order to select the optimum solution, we have to use some *a priori* knowledge as constraints such as smoothness and so on. In GA, the constraints are introduced in a simple manner. They are just associated as additional

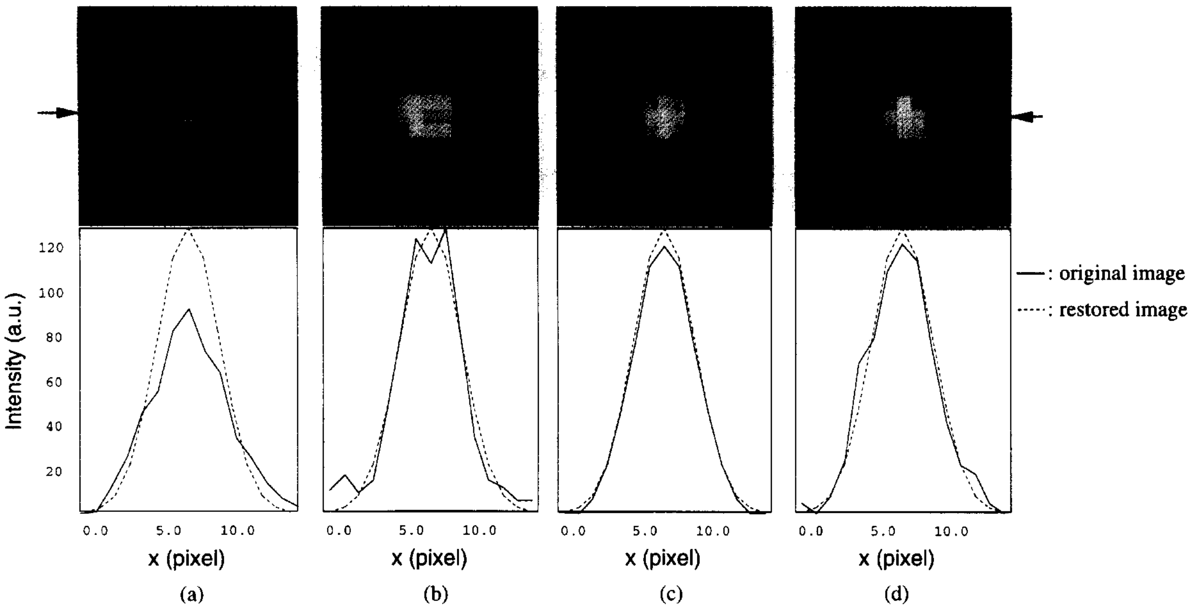


Fig. 4. Genetic restoration of a gray image. (a) Blurred image with an additive Gaussian noise, (b) restored image with Laplacian constrained ($\lambda = 3.0$) after 1000th generation, (c) restored image with Laplacian constrained ($\lambda = 3.0$) after 4500th generation, and (d) same as (c) but without constraint ($\lambda = 0.0$).

terms in the cost function. The newly proposed constrained cost function is shown in Eq. (10):

$$E = \|g - \hat{f} * h\|^2 + \lambda \cdot \|\boldsymbol{P} \cdot \hat{f}\|^2, \tag{10}$$

where the second term is the constraint. $\boldsymbol{P}$ is the constraint operator, and the Laplacian operator is used here to obtain a "smooth" estimate. The Laplacian value of pixel $f_{i,j}$ is calculated using surrounding four pixels ($\boldsymbol{P} \cdot f_{i,j} = \nabla^2 f_{i,j} = f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4f_{i,j}$). λ is a parameter used to balance the first term and the second term. Its value is roughly estimated as $M^2\sigma^2/\|\boldsymbol{P} \cdot g\|^2$, where M is the pixel number of the blurred image and σ^2 is the variance of noise contained in the blurred image.

4. Simulation results

We have carried out computer simulations to validate the applicability of GA for image restoration. Two typical phantoms are used in simulations. Fixed parameters for genetic algorithms, which are obtained through several test runs, are shown in Table 1. The probability = 0 means the corresponding crossover or mutation is not to be used.

The first object is "*E*" with 35 × 35 pixels, which is a binary image. It is blurred by the out of focusing (defocusing). The PSF of the defocusing is shown in Eq. (11):

$$h(x, y) = \begin{cases} 1/\pi a^2, & (x^2 + y^2) \leqslant a^2, \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where $a = 7$. Fig. 3a shows the blurred image with an additive Gaussian noise ($\sigma = 0.1, \mu = 0$), which is used as an input image. The restorations by the proposed GA are shown in Fig. 3b–e. Fig. 3b, c, d and e are produced with 0th, 200th, 400th, and 1000th generation, respectively. It can be seen that the restored image is improved with increase in the number of generation. In order to make a comparison, the restoration by a typical linear method called Wiener filter [11] is also shown in Fig. 3f. The image quality by the typical linear method is poor.

Since one of the purposes of this research is to restore a X-ray pinhole image of laser-imploded target which is blurred by a low-pass filter and significant pinhole size [4,9], the second test object as shown in Fig. 2a is used as a simulated laser-imploded target. There is a hot core (high intensity) at the

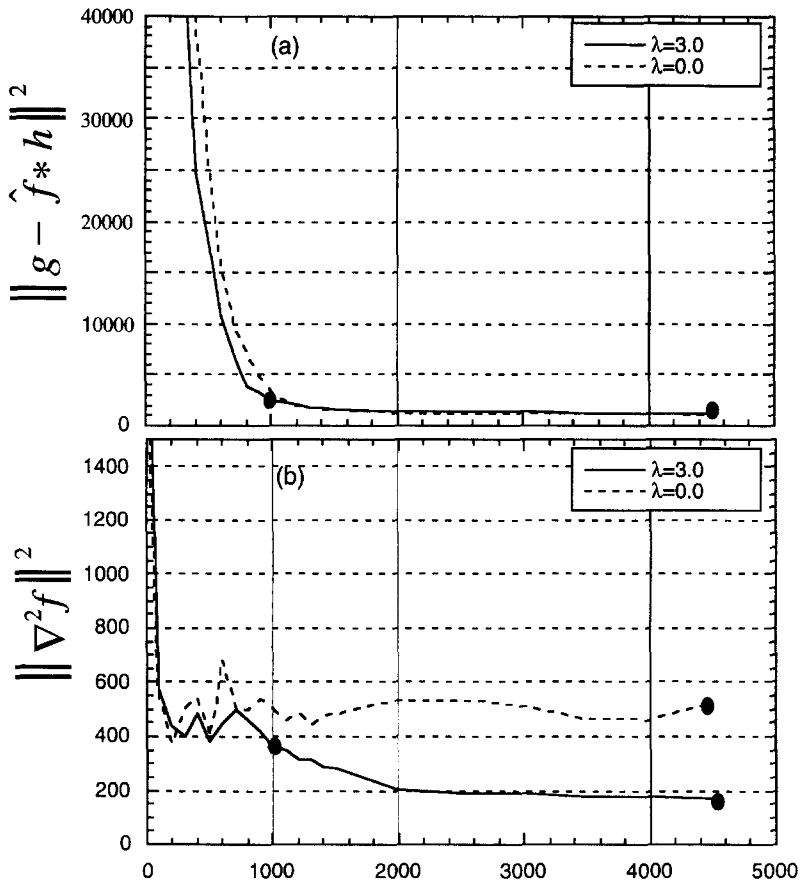


Fig. 5. Comparison of difference error (a) and Laplacian value of the restored image (b) with and without Laplacian constraint.

center surrounding by cold plasma (low intensity). The image is with 128 gray levels and the object size is 15 × 15. Fig. 4a is the blurred pinhole image obtained by convolving the object phantom with a pinhole. The pinhole size is 5 × 5. An additive Gaussian noise ($\sigma = 3\% \cdot I_{\max}, \mu = 0$) is also added to the blurred image, where $I_{\max}$ is the maximum intensity of the object phantom. The restorations by the constrained GA ($\lambda = 3$) after 1000th generation and 4500th generation are shown in Fig. 4b and c, respectively. And in order to make a comparison, we also show the restoration by the normal GA ($\lambda = 0$) in Fig. 4d. Fig. 5a and b show the difference error between the calculated blurred image ($\hat{f} * h$) and the obtained blurred image (g), and Laplacian value of the reconstructed images. Solid line is the result with constraint (Eq. (7)) and dashed line is the result without constraint (Eq. (2)). As shown in Fig. 5a and b, the difference errors for both methods are almost same, while the Laplacian values of the restored images are quite different, resulting in different restorations (Fig. 4c and d). Only the case with constraint, the Laplacian value is guided to that of the original image. It is clear that the restored image is improved by the constraint in comparison to that without constraint. Thus we can see that the constrained GA method is very tolerant of the noise. It should be noticed that since the inverse of the blurring function does not exist in this case, it is impossible to recover the original image from the blurred image using the linear techniques even in the absence of noise [12].

Fig. 6a shows a real X-ray pinhole image of laser-imploded target obtained in laser fusion experiments [9]. The pinhole size is about 12 μm corresponding to

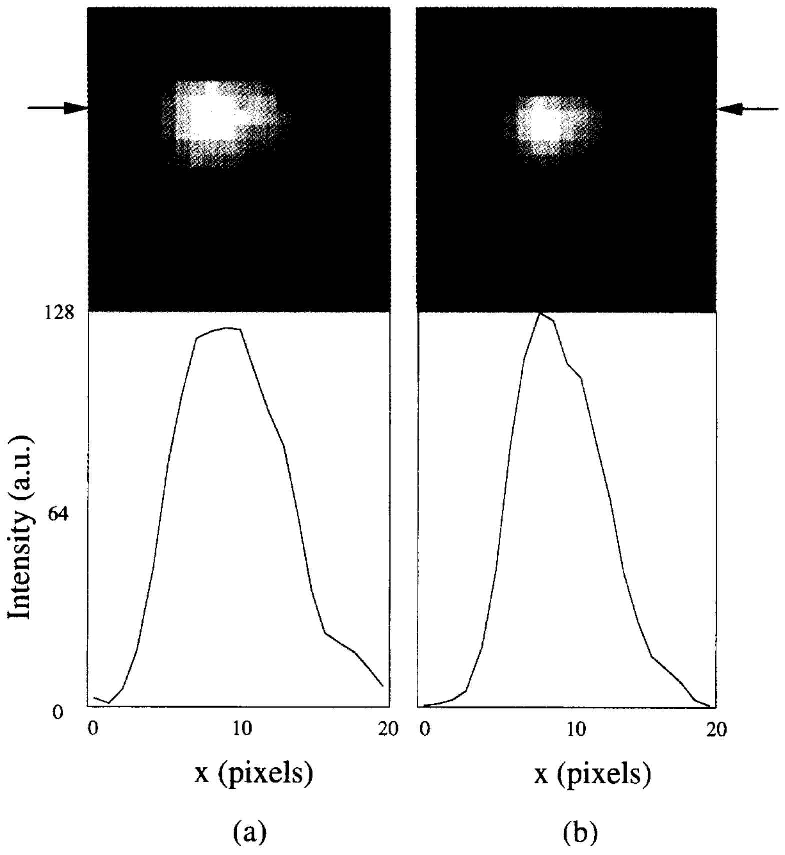


Fig. 6. Restoration of real X-ray pinhole images: (a) X-ray pinhole image of laser-imploded target, (b) restored image by the constrained GA after 4500th generation.

5 pixels on detector. The pinhole image was recorded by a X-ray CCD camera (Princeton Instrument) [9]. Fixed parameters for genetic algorithms is the same as that used for the second object as shown in Table 1. The restored image by the constrained GA after 4500th generation is shown in Fig. 6b. A shaper and smoother image is obtained. From the simulation results (Fig. 4) with the second object, the restored image (Fig. 6b) is more exact than that without restoration (Fig. 6a).

5. Parallel implementation

In the previous section, we have shown the great advantage of GA for image restoration. The disadvantage of GA is its very high computation cost (the simulation results shown in Section 4 required more than 1 h of CPU time on a 32-bit workstation IBM RS/6000).

In order to reduce the large computation cost, parallel implementation of GA for image restoration has been done [3] and its performance has been evaluated in a multi-workstation environment under the control of parallel virtual machine (PVM) [6], which is a free network connecting software developed by ORNL. The typical flowchart for the parallel implementation is shown in Fig. 7. We first create a population of estimates $\hat{f}$ of the original image as chromosomes randomly in the code and initialization process at a master workstation (WS). Then the chromosomes are divided into n subgroups, where n is the number of processors, and every subgroup is sent to its corresponding slave workstation. At a slave workstation, the degraded image $\hat{g}$ $(=\hat{f} * h)$ and the cost are calculated for each chromosome.

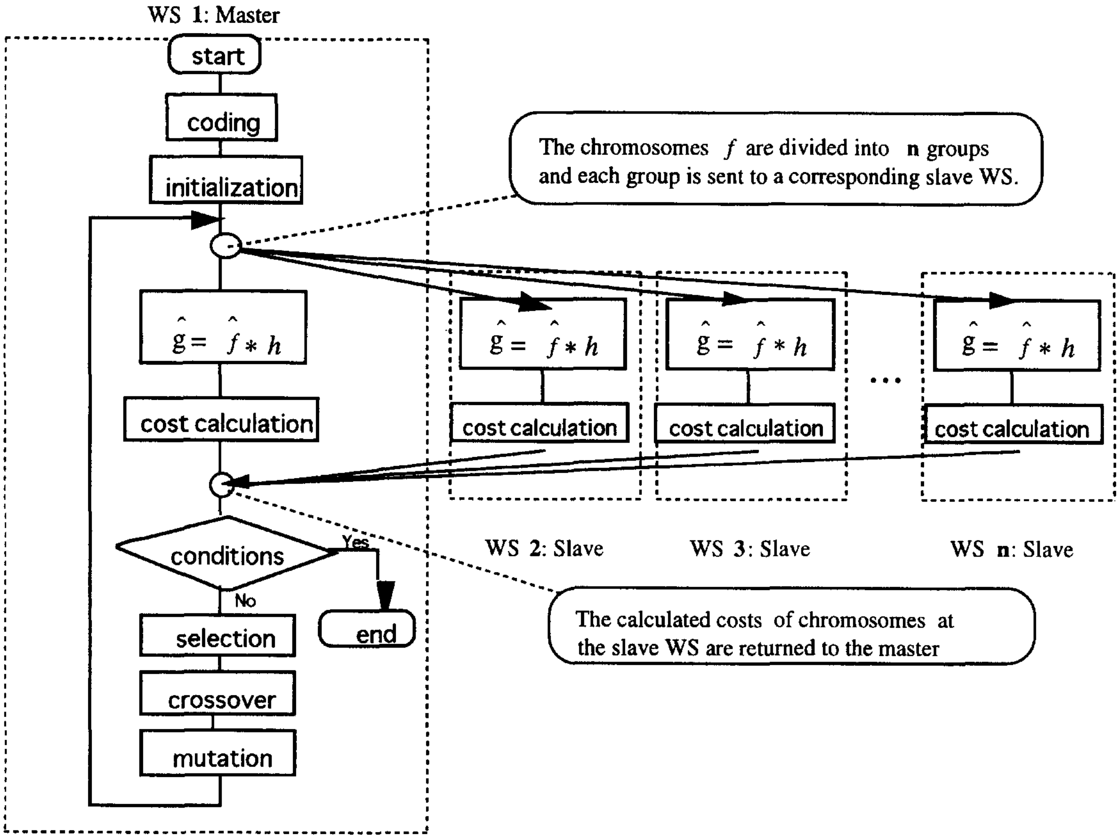


Fig. 7. Parallel implementation of GA for image restoration.

After the calculation of cost at the slave WS, the costs are then returned from the slave to the master, where three evolutionary operators (selection, crossover and mutation) are performed on the whole populations to guide the chromosomes toward an optimum solution (minimum cost).

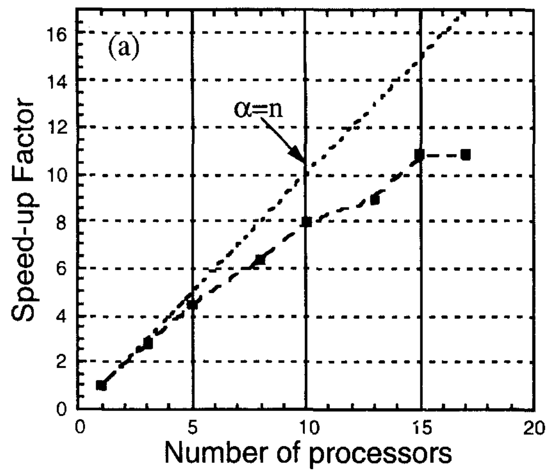
Let us define the speed-up factor α and parallel efficiency β as

$$\alpha = T_1/T_n, \qquad \beta = \alpha/n,$$

where T_1 and T_n are the total calculation time when the processor number is 1 and n, respectively. The calculated speed-up factor and parallel efficiency are shown in Fig. 8a and b, respectively. It can be seen that a speed-up factor of 10 can be achieved when we use 15 processors, corresponding to a parallel efficiency of 67%.

5. Summary

We have applied genetic algorithms (GA) with Laplacian constraint for restoration of gray images in presence of noise. The performance and potential of the proposed method have been demonstrated with computer simulations. The simulation results show that the restored image is improved in comparison to the restoration without Laplacian constraint. In order to reduce the large computation cost of GA, a prototype parallel implementation of GA-based restoration has been realized in a multi-workstation environment under the control of PVM. The large computation cost of the GA can be significantly reduced, and the reduction can be to 70–80% of the single processor computation cost. Further investigation is under way to develop more efficient genetic operators to realize restoration of large images.



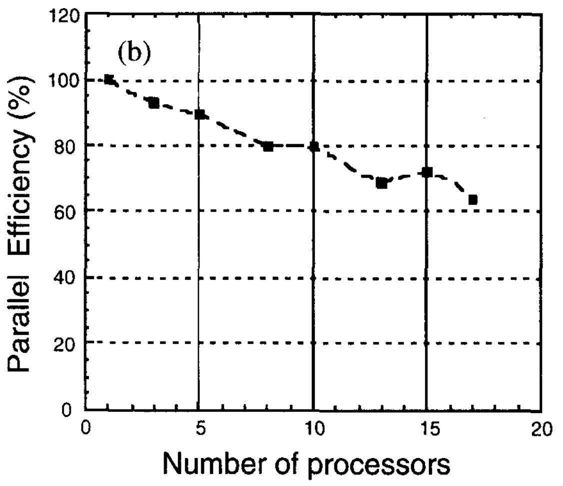


Fig. 8. (a) Speed-up factor vs. number of processors, (b) parallel efficiency vs. number of processors (original image: 15 × 15 pixels; PSF: 11 × 11 pixels).

Acknowledgements

The authors like to thank Mr. M. Iguchi for his helpful discussion on genetic algorithm.

References

[1] S.F. Burch, S.F. Gull, J. Skilling, Image restoration by a powerful maximum entropy method, Comput. Graphics Image Process. 12 (1983) 113–128.

[2] Y.-W. Chen, Z. Nakao, K. Arakaki, S. Tamura, Blind deconvolution based on genetic algorithms, IEICE Trans. Fundam. E-80-A (1997-12) 2603–2607.

[3] Y.-W. Chen, Z. Nakao, X. Fang, Parallelization of a genetic algorithm for image restoration and its performance analysis, Proc. of IEEE 3rd Int. Conf. on Evolutionary Computation, Nagoya, Japan, 1996, pp. 463–468.

[4] Y.-W. Chen, N. Miyanaga, M. Yamanaka, M. Nakai, T. Yamanaka, S. Nakai, Three-dimensional imaging of laser-imploded targets using computed tomography technique, IEEE Trans. NS-44 (1997) 890–893.

[5] B.R. Frieden, Restoring with maximum likelihood and maximum entropy, J. Opt. Soc. Amer. 62 (1972) 511–518.

[6] A.L. Geist, A. Beguelin, J. Dongarra, W. Jang, R. Manchek, V. Sunderam, PVM: Parallel Virtual Machine, A User's Guide and Tutorial for Networked Parallel Computing, MIT Press, New York, 1994.

[7] S. Geman, D. Geman, Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images, IEEE Trans. PAMI-6 (1984) 721–741.

[8] J. Maeda, K. Murata, Image restoration by an iterative regularized pseudoinverse method, Appl. Opt. 20 (1984) 857–861.

[9] T. Matushita, R. Kodama, Y.-W. Chen, M. Nakai, K. Shimada, M. Saito, Y. Kato, Imaging system using X-ray CCD cameras for laser fusion experiments, 1995 Annual Progress Report, ILE, Osaka University, 1996, pp. 183–185.

[10] Z. Michalewics, Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin, 1992.

[11] A. Rosenfeld, A.C. Kak, Digital Picture Processing, 2nd ed., Academic Press, New York, 1982.

[12] K. Takazu, H. Sawai, S. Watanabe, M. Yonayama, Genetic algorithms applied to Bayesian image restoration, IEICE Trans. J77-D-II (1994) 1168–1776 (in Japanese).