IPSJ Transactions on Mathematical Modeling and Its Applications Vol. 3 No. 3 154–165 (Oct. 2010)

Regular Paper

Effectiveness of Genetic Multistep Search in Unsupervised Design of Morphological Filters for Noise Removal

Yoshiko Hanada,[†1] Mitsuji Muneyasu[†1]
and Akira Asano[†2]

In this paper, the effectiveness of deterministic Multi-step Crossover Fusion (dMSXF) and deterministic Multi-step Mutation Fusion (dMSMF), which are types of genetic multistep searches based on a neighborhood search mechanism, in solving an unsupervised design problem of suitable structuring elements (SEs) of a morphological filter is shown. In our previous work, it was shown that dMSXF and dMSMF are very effective for solving combinatorial optimization problems, particularly on problems for which the landscape is an AR(1) landscape observed in the NK model. In addition, their effectiveness for reproduction mechanisms to obtain the offspring was shown to be retained with increasing level of epistasis. In this paper, we show that a characteristic of the AR(1) landscape is observed in an objective function for the unsupervised design of SEs, and superior search performances of both dMSXF and dMSMF for conventional crossover are shown. The processing results of the obtained SEs are also compared with those of conventional filters used for impulse noise removal.

1. Introduction

Images are often corrupted, blurred or distorted by various causes in digital equipment. Among them, impulse noise due to a noisy circuit or a channel transmission is known to significantly degrade the image quality, and it is important to find the impulse noise and to replace it with the estimated original value of the corrupted pixel [1]. Various filters such as median filters based on order statistics have been proposed [2)–6)]. As another way of recovering images from impulse noise, morphological filters based on mathematical morphology have been used [7)–9)]. Mathematical morphology is a fundamental framework for image manipulation, and a wide range of nonlinear image processing filters can be unified within this framework. *Opening* and *closing*, which is the dual of opening, are typical morphological operations and fundamental morphological filters that are idempotent. Mathematical morphological operations manipulate an image with a small object called a *structuring element* (SE), which is equivalent to the window of the image-processing filters. The opening operator composes the resultant image object by arranging an SE inside a target object and removes residual regions that are too small to hold the SE inside. The significance of opening is its quantitativeness with respect to the size of the image object. The removal of impulse noise by opening is quantitatively achieved in the sense that noise objects smaller than the SE are precisely removed. Since this operator composes an image by repeatedly locating an SE, its shape and grayscale distribution appear directly in the resultant image. When an SE is inappropriate for the image, it causes the appearance of undesired microstructures that are unrelated to the original image. This problem can be avoided by applying an appropriate grayscale SE that resembles the objects in the target image. Thus, the determination of both the shape and the grayscale distribution of an SE is an important problem. Here, we use the opening operator for noise removal in textures corrupted by impulse noise.

Evolutionary computations or neural networks have been applied to design suitable SEs or filters considering the characteristics of the target corrupted images [10)–15)]. These methods require reference images or test patterns that are similar to the target image, and the difference between it and the filtered image is minimized to optimize the SE or filter. Alternatively, interactive evolutionary computations are applied to evaluate the filtering performance in order to design filters [16)]. For practical use, we design, for textures, suitable SEs that remove noise and directly reconstruct the image with high accuracy from only the corrupted image. Our approach independently designs SEs, focusing on specific characteristics of the target texture. However, it can be applied to natural images through combination with a segmentation algorithm [17)].

In this paper, a genetic algorithm (GA) [18)] is adopted as the optimization algorithm for the unsupervised design. When we apply a GA to a problem, particularly a combinatorial problem, it is important to design a crossover method that

†1 Kansai University
†2 Graduate School of Engineering, Hiroshima University

emphasizes the heredity of the favorable characteristics of parents. Deterministic Multi-step Crossover Fusion (dMSXF) [19], which is a type of genetic multistep search based on neighborhood search mechanisms, is a promising crossover operator for combinatorial problems. dMSXF can be constructed by introducing both a problem-specific neighborhood structure and a distance measure. By multistep search, dMSXF can generate a wide variety of offspring between parents and it performs particularly well in problems for which the landscape is an *AR(1) landscape*, such as the Traveling Salesman Problem (TSP) [19]. Deterministic Multi-step Mutation Fusion (dMSMF) is a complementary search method to dMSXF and explores outside the distribution of the population. It has been shown that the incorporation of dMSMF into dMSXF improves the search performance in several combinatorial problems [20]. In addition, we have shown that the high search performance of both dMSXF and dMSMF was achieved by setting the neighborhood size to a value near the correlation length, which is an indicator of the level of epistasis [21]. In this work, dMSXF is adopted as the crossover method to optimize the SEs of morphological filters for texture images. First we introduce the unsupervised design based on the *Primitive, Grain and Point Configuration* (PGPC) texture model [22]. The landscape of the objective function used in the design of SEs is experimentally shown to be similar to an AR(1) landscape observed in the NK model [23]. In numerical experiments, we apply dMSXF to typical textures, and show the effectiveness of dMSXF as the main search operator of a GA. Then, the incorporation of dMSMF into dMSXF is shown to improve the performance in this design problem. The processing results of the obtained SEs are compared with those of conventional filters used for impulse noise removal.

2. Genetic Multistep Searches

GAs are among the most effective approximation algorithms for optimization problems. GAs are applicable to a wide range of problems and have been applied to numerous combinatorial problems, such as the TSP, and various scheduling problems. In combinatorial optimization problems, GAs actualize an effective search through the use of genetic operators for the inheritance and acquisition of characteristics.

To apply GAs to these problems, it is important to design a crossover method to consider problem-specific structures and characteristics. Various crossovers focusing on the inheritance of parents' characteristics have been discussed. Among them, Multi-step Crossover Fusion (MSXF) [24] and deterministic MSXF (dMSXF) [19], which are types of genetic multistep searches based on neighborhood search mechanisms, have been proposed, since the incorporation of neighborhood searches into GAs is essential in order to adjust the structural details of solutions in combinatorial problems [25]. dMSXF is an improved MSXF method and can be constructed by introducing both a problem-specific neighborhood structure and a distance measure.

The acquisition of characteristics that do not appear in the parents can be achieved by incorporating deterministic Multi-step Mutation Fusion (dMSMF) [20] into dMSXF. dMSMF also performs a multistep neighborhood search and explores the external domain of the population distribution.

2.1 Deterministic Multi-step Crossover Fusion

dMSXF performs a neighborhood search using a deterministic rule composed of only a distance measure in a problem-independent manner. It advances the neighborhood search from a parent p_1 in the direction approaching the other parent p_2. The procedure of dMSXF is as follows. A set of offspring generated by parents p_1 and p_2 is indicated by $C(p_1, p_2)$.

Procedure of dMSXF

0. Let p_1 and p_2 be parents and set their offspring as $C(p_1, p_2) = \phi$.
1. $k = 1$. Set the initial search point $x_1 = p_1$ and add x_1 into $C(p_1, p_2)$.
2. /Step k/ Prepare $N(x_k)$ composed of μ neighbors generated from the current solution x_k. $\forall y_i \in N(x_k)$ must satisfy $d(y_i, p_2) < d(x_k, p_2)$.
3. Select the best solution y from $N(x_k)$. Let the next search point x_{k+1} be y and add x_{k+1} into $C(p_1, p_2)$.
4. Set $k = k + 1$ and go to step 2 until $k = k_{\max}$ or x_k equals p_2.

At step 2 of the procedure of dMSXF, every neighborhood candidate y_i ($1 \leq i \leq \mu$) generated from x_k must be closer to p_2 than x_k. In addition, dMSXF necessarily moves its transition toward p_2 even if all solutions in $N(x_k)$ are inferior

Table 1 Application of dMSXF to the 1-max problem: $L_{bit} = 10$, $k_{\max} = 3$, $\mu = 3$.

p_1	*step*1	*step*2	p_2
(base solution x_k)	0111110000 (5)	0111110110 (7)	1101111110 (8)
0111110000 (5)	01$\hat{0}\bar{0}$11$\hat{1}$000 (4) 01$\hat{0}$1110$\hat{0}$10 (5) 0111110$\hat{1}\hat{1}$0 (7) →	$\hat{1}\bar{1}\hat{0}$1111$\hat{1}$110 (8) → $0\hat{0}$1$\hat{0}$110110 (5) $\hat{1}\hat{0}\bar{0}$110110 (6)	1000111110 (6)

$\bar{*}$ of x_k denotes the difference between x_k and p_2, and $\hat{*}$ is introduced from p_2.
$d(p_1, p_2)/k_{\max}(= 7/3 = 2$ or $3)$ bits are introduced from p_2 at each transition.

to the current solution x_k. dMSXF requires two parameters, $k_{\max}$ and μ. $k_{\max}$ is the number of steps in the neighborhood search and μ is the number of solutions generated at each step of the neighborhood search. In the procedure of dMSXF, at most $k_{\max} * \mu$ solutions are generated, and $C(p_1, p_2)$ is comprised of the best neighbor solutions, i.e., $\{x_1, x_2, \cdots, x_{k_{\max}}\}$.

Table 1 shows an example of the application of dMSFX to a 1-max problem. In this problem, the Hamming distance is adopted as the distance measure. The bits copied from p_2 to x_k are chosen randomly at each step. The step size of the neighborhood search is $d(p_1, p_2)/k_{\max}$ $(= w)$, and w bits are introduced from p_2 at each transition.

2.2 Deterministic Multi-step Mutation Fusion

In contrast to dMSXF, in which the search approaches the other parent, dMSMF advances the search in the direction away from the parents' neighborhood in a deterministic manner using both the quality of solutions and the distance measure as follows.

Procedure of dMSMF

0. Let p_1 and p_2 be parents and set their offspring as $C(p_1, p_2) = \phi$.
1. $l = 1$. Set the initial search point $x_1 = p_1$.
2. /Step l/ Prepare $N(x_l)$ composed of λ neighbors generated from the current solution x_l. $\forall y_i \in N(x_l)$ must satisfy both $d(y_i, p_1) > d(x_l, p_1)$ and $d(y_i, p_2) > d(x_l, p_2)$.
3. Select the best solution y from $N(x_l)$. Let the next search point x_{l+1} be y and add x_{l+1} into $C(p_1, p_2)$.
4. Set $l = l + 1$ and go to step 2 until $l = l_{\max}$.

At step 2 of dMSMF, every neighborhood candidate y_i $(1 \le i \le \lambda)$ generated from x_l must satisfy both $d(y_i, p_1) > d(x_l, p_1)$ and $d(y_i, p_2) > d(x_l, p_2)$. Even if all solutions in $N(x_l)$ are inferior to the current solution x_l, the transition to a solution in $N(x_l)$ is necessarily accepted. In this procedure, at most $l_{\max} * \lambda$ solutions are generated. Only in the case of binary problems is dMSMF a functionally equivalent operator to dMSXF between p_1 and $\hat{p_2}$, where $\hat{p_2}$ represents the solution consisting of, genetically, the complete reverse of p_2.

2.3 Generation Alternation Model

The generation alternation model we used for dMSXF or dMSXF+dMSMF in this paper is outlined below. This model focuses on the local search performance, and it have been shown to be effective in combinatorial optimization problems [19),20),26)].

Generation Alternation Model

0. Generate the initial population composed of N_{pop} random solutions (individuals) $\{x_1, x_2, \cdots, x_{N_{pop}}\}$.
1. Reset indexes $\{1, 2, \cdots, N_{pop}\}$ to each individual randomly.
2. Select N_{pop} pairs of parents (x_i, x_{i+1}) $(1 \le i \le N_{pop})$, where $x_{N_{pop}+1} = x_1$.
3. For each pair (x_i, x_{i+1}), if $d(x_i, x_{i+1})$ is greater than predefined value d_{min}, apply dMSXF; otherwise apply dMSMF.
4. For each pair (x_i, x_{i+1}), select the best individual c from offspring $C(x_i, x_{i+1})$ generated by parents (x_i, x_{i+1}) and replace the parent x_i with c.
5. Return to step 1 until some terminal criterion is satisfied, e.g., a number of generations and/or evaluations.

3. Unsupervised Design of Structuring Elements for Noise Removal

3.1 Mathematical Morphology and Opening

In the context of mathematical morphology, an image object is defined by a set. In the case of binary images, this set contains the pixel positions included in the object, i.e., those of white pixels. In the case of grayscale images, an image object is defined by an *umbra* set. If the pixel value distribution of an image object is denoted as $f(\boldsymbol{x})$, where $\boldsymbol{x} \in \mathbb{Z}^2$ is a pixel position, its umbra $U[f(\boldsymbol{x})]$ is defined as

$$U[f(\boldsymbol{x})] = \{(\boldsymbol{x}, t) \in \mathbb{Z}^3 | f(\boldsymbol{x}) \geq t > -\infty\}. \quad (1)$$

Consequently, when we assume a solid whose support is the same as a grayscale image and whose height at each pixel position is the same as the pixel value at this position, the umbra is equivalent to this solid and the whole volume below this solid within the support. Another object, called a *structuring element* (SE), is defined in the same manner as above. The SE is equivalent to the window of image-processing filters and is considered to be much smaller than the image object. *Opening* and *closing* are typical morphological operations and fundamental morphological filters that are idempotent. They are used for various methods of noise reduction and object extraction, for example. In the case of a binary image and an SE, the opening of an image object X with respect to an SE B, denoted X_B, has the following property:

$$X_B = \{B_z | B_z \subset X, z \in \mathbb{Z}^2\}, \quad (2)$$

where B_z indicates the translation of B by z.

Here, we concentrate on the opening, since the operations on closing are regarded as the dual of opening. In the case of the grayscale image and an SE, the opening is similarly defined by replacing the sets X and B with their umbrae, respectively, and assuming that $z \in \mathbb{Z}^3$. This property indicates that opening is the regeneration of an image produced by arranging the SE and removes smaller white regions than the SE in the binary case. In the case of grayscale image, the regions that are smaller than SE and have brighter pixels than those in their neighborhood are removed. The fact that the opening operator eliminates smaller structures and smaller bright peaks than the SE indicates that it acts as a filter to distinguish object structures by their size. In this paper, we adopt the opening with an SE as an impulse noise removal filter for texture images.

3.2 PGPC Texture Model and Primitive Estimation

When applying the opening operator to regenerate the target texture image, the most important issue is how to choose the best SE that describes the characteristics of the target. The *Primitive, Grain and Configuration* (PGPC) model [22] is one of the methods for describing a texture image, and it represents a texture as an image composed of a regular or irregular arrangement of objects much smaller than the size of the image. In this model, the objects arranged in the texture are called *grains*, and the grains are considered to be derived from one or a few typical objects called *primitives*. We assume that the primitive is expressed by an SE. We also assume here that the grains are derived from the r-fold homothetic magnification of one SE B, where the magnification is defined as the (r-1)-fold *Minkowski set addition* between SE B and another small element such as a single dot. The grain obtained as a result of this magnification is expressed as rB. In this case, X_{rB} is regarded as the texture image X composed by the arrangement of rB only. It follows that rB-$(r+1)B$ indicates the region included in the arrangement of rB but not in that of $(r+1)B$. Consequently, X_{rB}-$X_{(r+1)B}$ is the region where r-size grains are arranged if X is expressed by employing an arrangement of grains that are preferably large magnifications of the primitive. The sequence $\{X\text{-}X_B,\ X_B\text{-}X_{2B},\ \cdots,\ X_{rB}\text{-}X_{(r+1)B},\ \cdots\}$ is the decomposition of the target texture into the arrangement of grains of each size. Since the sequence can be derived by using any SE, it is necessary to estimate the appropriate primitive that is truly a typical representative of the grains.

In the estimation of a primitive (an SE) to regenerate precisely the target image, the candidate SE yielding the simplest arrangement by its sufficiently high homothetic magnifications is considered to be the optimal one. This concept is similar to the minimum description length (MDL) principle; the distribution of $X_{rB} - X_{(r+1)B}$ has a heavy tail in small values of r for the optimal SE in this case. In our unsupervised design approach, the principle of the PGPC model is also the basis for constructing the objective function used for the estimation of SEs. To remove impulse noise from the corrupted image, among the SEs with their magnifications that are considered to be satisfying the condition described above, an SE that can well eliminate instantaneous peaks is preferred.

3.3 Unsupervised Design of SEs

For textures corrupted by impulse noise, we design a suitable SE that can remove the noise and reconstruct the image accurately. Here, we consider grayscale texture images with a gray level of 256, and an SE that is expressed as a 2-dimensional shape B. Each element $e(i, j) \in B$ has a pixel value, i.e., a brightness, in the range [0, 255]. The pixel value of $e(i, j)$ is a design variable in this optimization problem. In our approach, for practical use, optimal SEs are designed directly from the corrupted image and no training images are required. In this section, we identify the problem and design the objective function.

3.3.1 Noise Model

Impulse noise is known to be formulated as a bit error arising in a noisy circuit or a channel transmission. There are several impulse noise models for images [4]. Here, we adopt the model below. In this model, $X_o(i,j)$ indicates the pixel values of the original image, and l represents a non-negative integer with a uniform distribution. $X(i,j)$ is rounded to 255 if $X_o(i,j)+l$ exceeds 255.

$$X(i,j)=\begin{cases} X_o(i,j)+l & \text{prob. } p \\ X_o(i,j) & \text{prob. } 1-p \end{cases} \tag{3}$$

3.3.2 Design of Objective Function

We adopt a *size distribution function* for constructing the objective function. This function has been shown to be effective in optimizing the shape and pixel values of SEs [27]. The opening result of image X with respect to SE B means that the residue of X is obtained by removing smaller structures than B, and it is denoted as X_B. We perform the opening of X with respect to the homothetic SEs rB, which are the result of r-fold homothetic magnification of B, and obtain the image sequence $\{X, X_B, X_{2B}, \cdots, X_{rB}, \cdots\}$. In this sequence, X_{rB} is obtained by removing the regions smaller than rB. In the case of grayscale images, at each r, we calculate the ratio of the value at each pixel position of X_{rB} to that of the original X, and then calculate the sum of the ratio [⋆1]. The function from size r to the corresponding ratio decreases monotonically and becomes unity when the size is 0. This function is called the size distribution function. The size distribution function of size r, $F(r)$, indicates the area ratio of the regions with sizes greater than or equal to r. Here, the integral (summation) of $F(r)$ from 1 to M shown in Eq. (4) is used as the objective function and SE B is optimized by minimizing it. This is because a positive correlation is observed between the evaluation values and the preferable processing results, as shown in the next section. In Eq. (4), N indicates the size of the target image, and $X_{rB}(i,j)$ and $X(i,j)$ respectively denote the values at the pixel position (i,j) in X_{rB} and X. No constraints are required in this optimization problem.

⋆1 In the original definition of the size distribution function, the ratio of the sum of the pixel values of X_{rB} to that of the original X is calculated. To consider the variance in the value among pixels, we calculate the ratio at each pixel value.

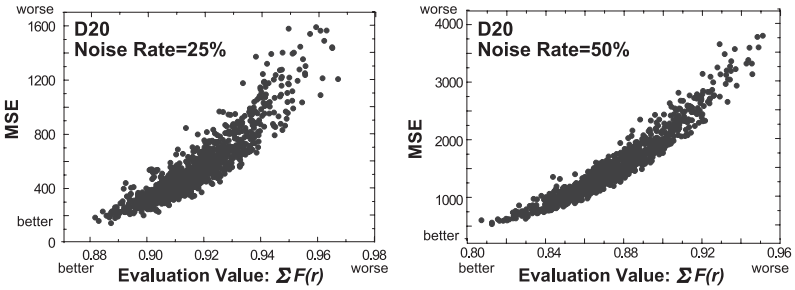


Fig. 1 MSE vs Evaluation Value in pixel value optimization.

$$f_{objective}=\sum_{r=1}^{M}F(r), \quad F(r)=\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{X_{rB}(i,j)}{X(i,j)} \tag{4}$$

M is the parameter that determines the size of the largest magnification of SE, and its setting depends on the characteristics of the target texture. $M=1\sim3$ has been shown to be effective for small textures [27]. Here, we adopt $M=1$ to reduce the calculation cost since we empirically confirmed that the filtering performances of obtained SEs were not different much among $M=1$, 2 and 3.

3.3.3 Validity of Objective Function

In this paper, we compare the filtering performance using the mean square error (MSE) between the original image X_o and the filtered image obtained by applying an SE to the corrupted image X.

Figure 1 shows the relationship between the evaluation value and the MSE of an SE in the corrupted texture D20. This texture is an example from the set of Brodatz textures [28] and is described in detail in Section 5. The distribution was derived from 500 SEs with pixel values randomly assigned in the range of [0, 255] to a 3x3 square shape. To generate corrupted images, 25% or 50% of the pixels in the original images with positions that were randomly selected were replaced with impulse noise.

From this result, at both noise rates, the distribution extends diagonally from the bottom left to the top right, and we can confirm experimentally that processing results closer to the original textures are obtained by minimizing the summation of $F(r)$.

3.4 Application of GA to the Design of SEs

An SE is an *individual* (candidate solution) of a GA. For simplicity, we fix the shape of every SE to a full square of 3x3 pixels and only optimize the pixel values of elements $e(i, j)$ $(1 \leq i, j \leq 3)$ in this square. For the initial population, a random value in the range [0, 255] is assigned to each element of an SE. During the search using the GA, the value of each element is coded to a binary string of 8 bits by *binary coding*; consequently, an individual of the GA is expressed as a binary string of 72 bits (= 8 bits * 9 elements). This design problem is a binary problem; therefore, dMSXF and dMSMF are implemented as the bitwise operator shown in Table 1.

4. Problem Difficulty Analysis and NK Model

4.1 Properties of Fitness Landscape

It is essential to investigate the problem difficulty before applying a GA to an individual problem. There are several measurements for evaluating the complexity in the fitness landscape of an objective function. Epistatic interactions among design variables affect the features of local landscapes. The degree of fitness correlation among neighborhood solutions reflects the level of epistasis, which appears as the local ruggedness in a fitness landscape. The *correlation length* ℓ_c is an indicator of the level of epistasis [29], and it is derived from the *random walk correlation function* $r(s)$ in Eq. (5), where a time series $\{f(x_t)\}$ defines the correlation between two points s steps away along a random walk of length m through the fitness landscape. $\bar{f}$ and σ_f^2 respectively denote the average and variance of the fitness values.

$$r(s) = \lim_{m\to\infty} \frac{1}{\sigma_f^2(m-s)} \sum_{t=1}^{m-s} (f(x_t) - \bar{f})(f(x_{t+s}) - \bar{f})$$
$$\ell_c = -1/\ln(|r(1)|) \tag{5}$$

If a time series $\{f(x_t)\}$ is *isotropic, Gaussian* and *Markovian*, then the function $r(s)$ is of the form $r(s) = r(1)^s = e^{-s/\ell_c}$. In this case, the landscape is called an *AR(1) landscape*. AR(1) landscapes have been found in various combinatorial optimization problems, such as the TSP, and can be created by the NK model. This landscape feature is also observed in the objective function used for optimizing SEs, as demonstrated later in Section 4.3.

The lower the value of ℓ_c, the more rugged the landscape. No fitness correlation is observed between the current point p and a point with a distance from p of greater than ℓ_c i.e., it is difficult to estimate the behavior of a point distant from p, whereas a neighbor within distance ℓ_c has a certain correlation with p [29]. The indicator ℓ_c has tremendous impact on the performance of any neighborhood search method, and we can obtain useful information from the current solution to generate neighborhood solutions if the scope of the neighborhood is smaller than ℓ_c.

4.2 Description of the NK Model

The NK model [23] is a simple and flexible fitness function model in which the ruggedness of the landscape can be tuned by changing one parameter. It is often used for analyzing the search behavior of a GA in problems involving AR(1) landscapes. In addition, various types of combinatorial optimization problems, such as the TSP, can be traced to the NK model in terms of problem structures and local properties of the landscape [30]. The NK model is represented by a binary string of length N. The parameter K represents the number of linkages each gene has to other genes in the same chromosome; the fitness contribution f_i of each gene x_i in chromosome x depends on x_i and the values of K other genes. For each gene x_i, each of the 2^{K+1} allele combinations is assigned a random real number value between [0, 1), and f_i is derived as the pregiven random value corresponding to the bitstring consisting of x_i and $K/2$ other genes. The fitness function of the NK model is the average overall fitness contribution described below. In this work, we maximize the fitness in the NK model.

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x_i, x_{i_1}, \cdots, x_{i_k}) \tag{6}$$

The parameter K, which is set to be from 1 to (N-1), indicates the level of epistasis and modifies the ruggedness of the landscape. With increasing K, the landscape becomes more rugged and the fitness correlation between neighborhood solutions also becomes smaller. There are two typical models of epistasis: the nearest-neighbor model and the random epistatic model [31]. In this work, we adopt the former, for which the bit x_i is linked to the $K/2$ nearest bits on each

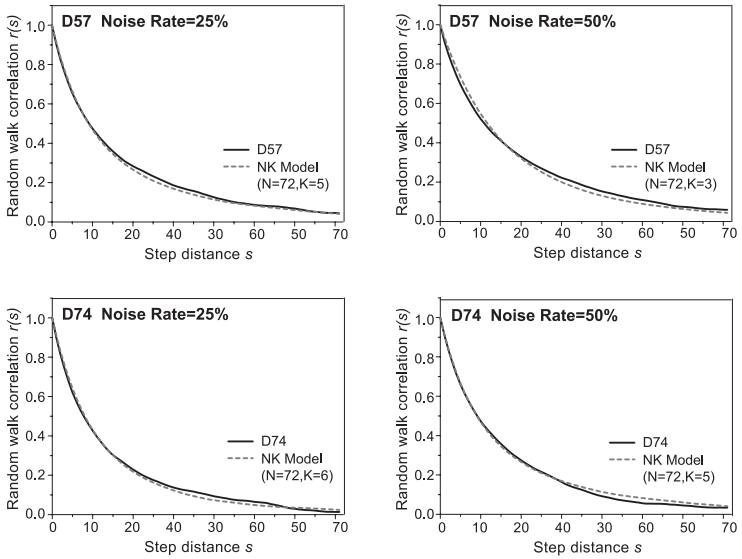


Fig. 2 Random walk correlation functions for texture images.

side of x_i. The fitness landscape of the NK model has the AR(1) landscape; the value of ℓ_c has been proved to be $N/(K+1)$ [32].

4.3 Fitness Landscape Feature of the Objective Function of SEs

Here, we show that the objective function used to design SEs for noise removal, that is, the summation of the size distribution, has the feature of an AR(1) landscape. Random walk correlation functions of this design problem were examined to analyze aspects of the fitness landscape and problem difficulty. At each step of the random walk, a neighborhood solution was generated by the bit flip operator. Two images, D57 and D74 from the set of Brodatz textures [28], were used for the examination. These images are described in detail in Section 5. The probability of impulse noise was set to 25% or 50% for both images.

The computed correlation functions are plotted in **Fig. 2**. These functions were estimated experimentally by performing random walks of length $1\text{x}10^5$. Random walk correlation functions for different parameters of the NK model are also plotted in the figure for comparison. All functions here are 72-bit problems, which is the same bit length as that used to represent the individuals used for designing SEs.

From the figure, we can see that the random walk correlation functions for the design problem of SEs have an exponentially decaying form as expected for the landscape of the NK model. This property was also observed in other values of M. The local ruggedness of the objective functions of these textures in the optimization of SEs is considered to be extremely similar to that in the NK model. It is observed that the ruggedness of the function is different for the two textures. In these examples, D74 is more rugged than D57. In addition, the level of epistasis of the objective function decreases with increasing noise rate.

It has been shown that dMSXF performs very well on an AR(1) landscape [21]. Therefore, it is expected that dMSXF will be effective used for optimizing SEs for noise removal. In addition, we can preliminarily examine the appropriate setting parameters of dMSXF using the NK model, before applying it to design SEs.

5. Numerical Experiments

We next discuss the suitability of dMSXF and dMSXF+dMSMF for application to the design of SEs. To show their effectiveness, we compare them with the conventional crossover operator, the uniform crossover (UX), which generates mostly intermediate offspring between parents in any rugged landscape. The processing performance of the SEs designed for impulse noise removal is also examined by comparing it to those of other typical filters.

Eight images, Brodatz textures [28] D3, D20, D57, D74, D87, D98, D101 and D112, which have a wide variety of shape characteristics and brightness, were used for the examination. The size of each image is 128x128 and the gray level of each image is 256. These textures are illustrated in **Fig. 3**.

Before applying dMSXF and dMSXF+dMSMF to the design of SEs, their search performance in the NK model is examined to determine the parameter settings at which they are effective. The efficacy of generating offspring is also compared using the progress rate (PR) [33], which is of the indicators for determining the efficiency of a search method. PR is the expectation value of the progress per one step under a given condition. The original PR is a measure of performance, particularly for the evolutionary strategy (ES) [34], and we adopt a

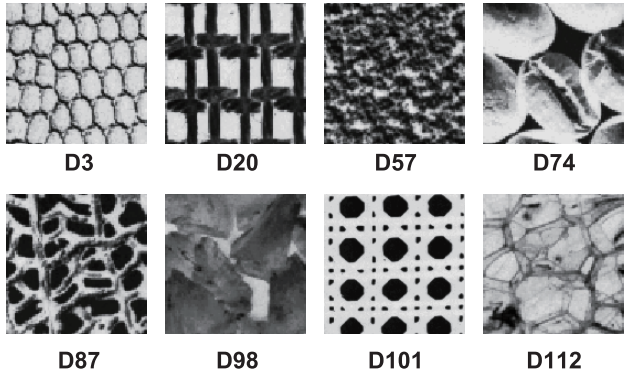


Fig. 3 Brodatz textures.

redefined PR for crossover methods [19]. The PR we adopt here is the average amount of progress from the better of the parents to the best of the family, as defined in Eq. (7). In Eq. (7), F is the family $F = \{p_1, p_2, C(p_1, p_2)\}$.

$$PR(p_1, p_2) = E(\min_{x \in F} f(x) - \min_{x \in \{p_1, p_2\}} f(x)) \tag{7}$$

5.1 Performance of dMSXF against the Level of Epistasis

We confirm the appropriate setting of parameters of dMSXF using examples of the NK model with $N = 72$. Since dMSMF searches a different solution space from that of dMSXF, but is a functionally equivalent operator to dMSXF in the case of binary problems, it is sufficient to examine the appropriate setting of parameters in dMSXF only. The effectiveness of dMSXF+dMSMF was described in our previous work [20),21]. In the experiments, the population size was set to 10. The generation alternation model based on the elitist recombination model (ER) [35] was adopted for UX.

Each trial was terminated after 25 generations in dMSXF and 50 generations in UX in all instances of the NK model. The total number of evaluations was the same for dMSXF and UX. For dMSXF, $k_{\max}$ was set to 2, 4, 8 or 16. The number of offspring generated by each pair of parents, N_C, was set to 32 for both methods. μ was calculated as $N_C/k_{\max}$.

Table 2 shows the search performances of the NK model with $N = 72$, where K was tuned in the range of 3 to 12. These results show the average and standard deviation of fitness in 100 trials. Larger values are considered to be better. At each trial, the same initial population was used in the GAs. **Figure 4** illustrates the PRs of both dMSXF and UX. In the figure, a white dot indicates the PR of UX and the black dots are the PRs of dMSXF. Larger values are also superior in this figure. These results show the average value of 1000 random pairs of parents p_1 and p_2, for which $d(p_1, p_2)$ is 9, 18 or 36 bits. The maximum distance, 36 bits, is derived from $N/2$, which is the average distance between parents in the initial population. The distances 9 and 18 are found in the middle stage of the search.

From Table 2, dMSXF shows a markedly superior performance to UX at all

Table 2 Search performances of dMSXF and UX ($N = 72$).

Parameters		$k_{\max}$=2		$k_{\max}$=4		$k_{\max}$=8		$k_{\max}$=16		UX=32	
K	ℓ_c	avg.	std.	avg.	std.	avg.	std.	avg.	std.	avg.	std.
3	18.0	0.908	0.028	0.909	0.032	0.905	0.030	0.888	0.032	0.884	0.035
6	10.3	0.903	0.036	0.908	0.038	0.895	0.037	0.869	0.041	0.879	0.035
9	7.2	0.736	0.022	0.738	0.023	0.739	0.026	0.723	0.024	0.728	0.025
12	5.5	0.694	0.022	0.702	0.016	0.700	0.018	0.687	0.017	0.690	0.019

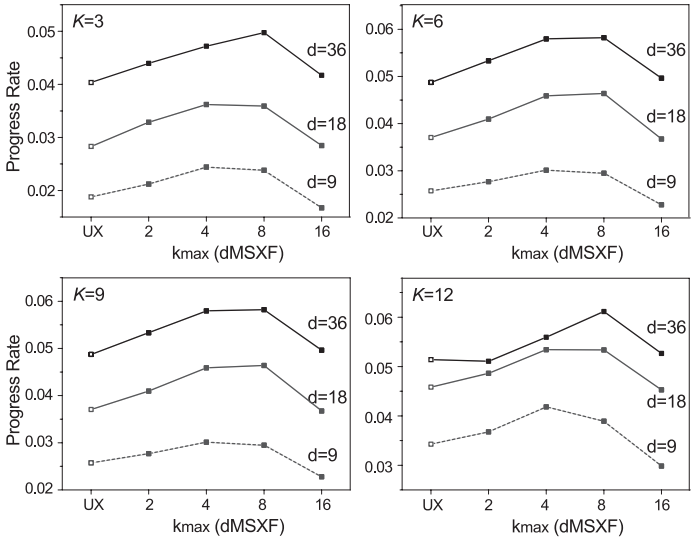


Fig. 4 Progress rate of crossovers.

levels of epistasis. Among the settings of dMSXF, $k_{\max} = 4$ performs satisfactorily in these examples. For the distance between parents observed in the initial generation, i.e., $d = 36$, we can see that $k_{\max} = 8$ generates offspring more effectively than other values of $k_{\max}$ from Fig. 4. On the other hand, when the population approaches convergence and the distance between parents becomes smaller, $k_{\max} = 4$ becomes more productive.

In these experiments, the total number of offspring N_C is fixed, and μ decreases with increasing $k_{\max}$. When the offspring are sufficiently large, dMSXF with a large $k_{\max}$ outperforms that with a small $k_{\max}$; however, it is preferable to obtain SEs with a small calculation cost, and we adopt $k_{\max} = 4$ when designing SEs.

5.2 Effectiveness of dMSXF in the Design of SEs

Here, we apply dMSXF to the optimization of SEs that are effective for impulse noise removal for textures. Eight images were used for the examination, and the probability of impulse noise was set to 25% or 50% for each image. The parameters of the GAs were the same as those described in the previous section. In dMSXF, $k_{\max}$ was set to 4, which is considered the most productive.

Table 3 shows the best value of MSE between the processing results and the original image, as well as the worst MSE (wst.), the average MSE (avg.) and the standard deviation of MSE (std.) out of 20 trials. Smaller values are considered to be better. The correlation length ℓ_c for each image, given in Table 3, was estimated experimentally by performing random walks of length $1\text{x}10^5$. Examples of estimations with the best solutions of dMSXF are shown in **Fig. 5**.

From Table 3, dMSXF performs better than the conventional crossover and yields good solutions. A marked improvement in processing performance is found in the worst MSE, which indicates that dMSXF can design suitable SEs stably.

Table 4 shows the comparison of the processing performances between the designed SEs and conventional filters: the median filter (MED) [1], the progressive switching median filter (PSM) [5] and the progressive switching weighted average filter (PSWA) [6] that are typical deterministic filters based on order statistics for impulse noise removal. A full 3x3 square window was adopted in these conventional filters. The processing performances of the SEs designed by GAs (dMSXF and UX) are also compared with that of the SE obtained by our previous unsupervised design method [27] using a simulated annealing [36], Mor(SA), in Table 4. The

Table 3 Processing results (MSE): dMSXF and UX.

Noise Rate = 25%										
Example			**dMSXF**				**UX**			
#	MSE	ℓ_c	best	wst.	avg.	std.	best	wst.	avg.	std.
D3	1506.4	11.2	260.6	356.6	292.0	21.2	266.7	438.0	294.8	35.7
D20	3052.3	9.7	105.9	245.0	121.5	30.1	106.4	239.6	133.9	37.3
D57	3996.3	10.2	274.5	362.8	287.9	24.1	275.4	439.6	312.1	53.8
D74	2102.7	9.4	110.3	132.8	118.0	6.4	111.9	136.2	121.3	6.5
D87	3298.5	10.6	280.7	311.1	291.0	6.5	287.0	336.9	299.4	12.5
D98	3712.0	9.1	27.7	36.7	30.4	2.4	27.7	52.1	32.9	6.8
D101	1819.3	9.3	261.3	580.6	354.9	95.6	282.9	688.3	397.9	114.3
D112	1682.6	9.6	87.7	105.7	93.2	5.2	88.5	130.9	99.4	10.6
Noise Rate = 50%										
Example			**dMSXF**				**UX**			
#	MSE	ℓ_c	best	wst.	avg.	std.	best	wst.	avg.	std.
D3	2918.7	11.6	441.3	505.3	462.7	18.9	443.1	543.2	471.5	31.3
D20	6331.9	11.4	265.3	319.4	273.1	13.2	266.4	331.7	277.2	14.5
D57	7979.4	11.7	383.7	442.4	392.3	12.7	384.9	463.7	404.9	23.7
D74	4076.7	10.9	226.2	286.2	237.5	13.4	226.4	289.3	252.4	22.0
D87	6580.7	11.6	472.4	516.4	486.3	12.2	473.0	529.8	491.3	15.0
D98	7409.7	9.6	69.5	76.3	71.7	1.8	70.1	123.6	84.7	13.6
D101	3667.0	11.1	438.9	513.8	462.4	20.1	440.2	533.7	470.8	24.8
D112	3427.6	10.9	180.7	230.3	188.6	10.5	181.9	256.4	200.9	20.7

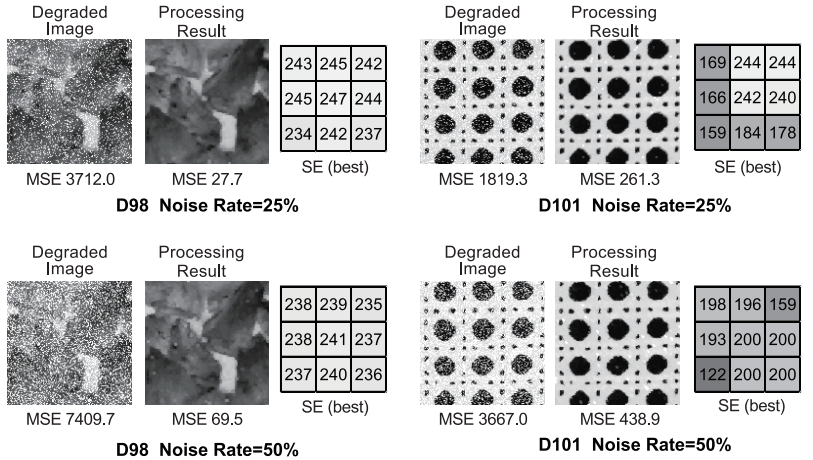


Fig. 5 The processing results obtained by dMSXF.

Table 4 Processing results (MSE): dMSXF, UX and other methods.

Noise Rate = 25%								
Example	**dMSXF**		**UX**		**Mor(SA)**	**MED**	**PSM**	**PSWA**
	bst.	wst.	bst.	wst.				
D3	260.6	356.6	266.7	438.0	340.5	919.6	608.5	490.0
D20	105.9	245.0	106.4	239.6	187.6	545.3	299.9	245.2
D57	274.5	362.8	275.4	439.6	288.4	778.5	341.6	299.1
D74	110.3	132.8	111.9	136.2	199.3	483.9	257.1	210.0
D87	280.7	311.1	287.0	336.9	380.0	896.0	496.7	402.8
D98	27.7	36.7	27.7	52.1	119.6	263.7	48.5	43.2
D101	261.3	580.6	282.9	688.3	255.7	688.8	416.6	344.7
D112	87.7	105.7	88.5	130.9	122.4	397.2	163.4	145.2
Noise Rate = 50%								
Example	**dMSXF**		**UX**		**Mor(SA)**	**MED**	**PSM**	**PSWA**
	bst.	wst.	bst.	wst.				
D3	441.3	505.3	443.1	543.2	713.3	2517.6	1505.9	773.7
D20	265.3	319.4	266.4	331.7	873.8	2878.1	1080.8	562.2
D57	383.7	442.4	384.9	463.7	807.3	3525.1	1194.7	537
D74	226.2	286.2	226.4	289.3	662.9	2038.5	826.8	382.5
D87	472.4	516.4	473.0	529.8	857.7	3230.1	1296.7	701.8
D98	69.5	76.3	70.1	123.6	429.2	2726.5	546.5	87.9
D101	438.9	513.8	440.2	533.7	788.4	2276.0	1260.8	458.2
D112	180.7	230.3	181.9	256.4	491.3	2127.5	878.5	257.4

Table 5 Processing results (MSE): dMSXF+dMSMF and dMSXF.

Noise Rate = 25%										
Example			**dMSXF+dMSMF**				**dMSXF**			
#	MSE	ℓ_c	best	wst.	avg.	std.	best	wst.	avg.	std.
D3	1506.4	11.2	257.9	309.2	287.0	15.6	260.6	356.6	292.0	21.2
D20	3052.3	9.7	105.0	148.3	113.7	13.8	105.9	245.0	121.5	30.1
D57	3996.3	10.2	271.5	323.7	280.9	14.0	274.5	362.8	287.9	24.1
D74	2102.7	9.4	109.1	124.4	115.9	3.5	110.3	132.8	118.0	6.4
D87	3298.5	10.6	280.2	299.3	288.4	4.4	280.7	311.1	291.0	6.5
D98	3712.0	9.1	27.6	32.3	28.7	1.1	27.7	36.7	30.4	2.4
D101	1819.3	9.3	256.3	557.2	345.4	96.5	261.3	580.6	354.9	95.6
D112	1682.6	9.6	87.3	95.6	89.3	1.7	87.7	105.7	93.2	5.2
Noise Rate = 50%										
Example			**dMSXF+dMSMF**				**dMSXF**			
#	MSE	ℓ_c	best	wst.	avg.	std.	best	wst.	avg.	std.
D3	2918.7	11.6	441.0	482.3	455.1	10.1	441.3	505.3	462.7	18.9
D20	6331.9	11.4	264.6	268.2	266.3	0.8	265.3	319.4	273.1	13.2
D57	7979.4	11.7	380.8	401.1	387.7	4.0	383.7	442.4	392.3	12.7
D74	4076.7	10.9	225.8	234.6	230.1	2.1	226.2	286.2	237.5	13.4
D87	6580.7	11.6	472.3	486.0	478.6	4.3	472.4	516.4	486.3	12.2
D98	7409.7	9.6	68.7	74.0	70.1	1.2	69.5	76.3	71.7	1.8
D101	3667.0	11.1	431.8	470.5	459.5	10.5	438.9	513.8	462.4	20.1
D112	3427.6	10.9	179.9	194.2	183.5	4.2	180.7	230.3	188.6	10.5

setting of parameters of Mor(SA) were the same as those described in Ref. 27). The results are the best values out of 20 trials.

From the comparison between the GAs and conventional methods, it is shown that our unsupervised approach can design effective SEs that remove the noise and reconstruct images with high accuracy. In particular, in the results for a noise rate of 50%, even the worst MSE of GAs is smaller than those of conventional methods for most textures.

5.3 Effectiveness of dMSXF+dMSMF in the Design of SEs

We examined the effectiveness of incorporating dMSMF into dMSXF. The same examples were applied for this examination. The values of the parameters of the GA were the same as those described in the previous section. In dMSXF+dMSMF, dMSMF was applied instead of dMSXF when the distance between parents was smaller than N x 0.1, where N denotes the bit length of the chromosome. The number of steps in the neighborhood search of dMSMF, $l_{\max}$, was set to 4, which is the same value as $k_{\max}$.

Table 5 shows the best value of MSE between the processing results and the original image, as well as the worst MSE (wst.), the average MSE (avg.) and the standard deviation of MSE (std.) out of 20 trials.

As can be seen from Table 5, dMSMF improves the design accuracy of SEs. From the perspective of both the average and the standard deviation of MSE, it is shown that dMSXF+dMSMF yields suitable SEs more stably than dMSXF, and we can confirm the importance of the search mechanism in exploring outside the distribution of the population in this design problem.

6. Conclusions

Deterministic Multi-step Crossover Fusion (dMSXF) and deterministic Multi-step Mutation Fusion (dMSMF) are superior genetic multistep operators for the inheritance and acquisition of characteristics and are particularly effective for problems associated with AR(1) landscapes. In this work, dMSXF and dMSMF were adopted as crossover methods for optimizing the structuring elements (SEs)

of morphological filters for texture images. A feature of an AR(1) landscape was observed in the objective function for the design of SEs, and it was shown that dMSXF and dMSXF+dMSMF can estimate more suitable SEs stably. SEs obtained by these operators also outperformed promising conventional filters. Here, we applied our unsupervised design method to the case of non-negative integer impulse noise; however, it is extendable to the combination of opening and closing to suppress more practical salt-and-pepper noise. This task is left as a future goal.

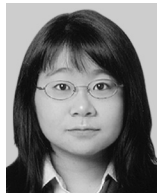
Acknowledgments This work was partially supported by Grant-in-Aid for Young Scientists (B), 22700158 of the Japan Society for the Promotion of Science (JSPS).

References

1) Pitas, I. and Venetsanopoulos, A.N.: *Nonlinear Digital Filters*, Kluwer Academic Publishers, Boston (1990).
2) Brownrigg, D.R.K.: The Weighted Median Filter, *Comm. ACM*, Vol.27, pp.870–818 (Aug. 1984).
3) Sun, T. and Neuvo, Y.: Detail-preserving Median Based Filters in Image Processing, *Pattern Recognition Letters*, Vol.15, No.4, pp.341–347 (1994).
4) Abreu, E., Lightstone, M., Mitra, S.K. and Arakawa, K.: A New Efficient Approach for the Removal of Impulse Noise from Highly Corrupted Images, *IEEE Trans. Image Processing*, Vol.5, No.6, pp.1012–1025 (1996).
5) Wang, Z. and Zhang, D.: Progressive Switching Median Filter for the Removal of Impulse Noise from Highly Corrupted Image, *IEEE Trans. Circuits & Systems II: Analog & Digital Signal Processing*, Vol.46, No.1, pp.78–80 (1999).
6) Matsumoto, T. and Taguchi, A.: Removal of Impulse Noise from Highly Corrupted Images by using Noise Position Information and Directional Information of Image, *Proc. Conference on Nonlinear Image Processing and Pattern Analysis XII. SPIE*, pp.188–196 (2001).
7) Serra, J.: *Image Analysis and Mathematical Morphology*, Academic Press (1982).
8) Soille, P.: *Morphological Image Analysis, 2nd Ed.*, Springer (2003).
9) Maragos, P.: Morphological Filtering for Image Enhancement and Feature Detection, *The Image and Video Processing Handbook*, Elsevier Academic Press, pp.135–156 (2005).
10) Harvey, N.R. and Marshall, S.: The Use of Genetic Algorithms in Morphological Filter Design, *Signal Processing: Image Communication*, Vol.8, pp.55–71 (1996).
11) Oh, J. and Luis F. Chaparro, L.F.: Adaptive Fuzzy Morphological Filtering of Impulse Noise in Images, *Multidimensional Systems and Signal Processing*, Vol.11, No.3, pp.233–256 (2000).
12) Rosin, P. and Hervas, J.: Image Thresholding for Landslide Detection by Genetic Programming, *Analysis of multi-temporal remote sensing images*, pp.65–72, World Scientific (2002).
13) Rosin, P.: Training Cellular Automata for Image Processing, *IEEE Trans. Image Processing*, Vol.15, No.7, pp.2076–2087 (2006).
14) Quintana, M.I, Poli, R. and Claridge, E.: Morphological Algorithm Design for Binary Images using Genetic Programming, *Genetic Programming and Evolvable Machines*, Vol.7, No.1, pp.81–102 (2006).
15) Faro, A., Giordano, D., Scarciofalo, G. and Spampinato, C.: Bayesian Networks for Edge Preserving Salt and Pepper Image Denoising, *Proc. Image Processing Theory, Tools & Applications*, pp.1–5 (2008).
16) Katsuyama, Y. and Arakawa, K.: Impulsive Noise Removal in Color Image Using Interactive Evolutionary Computing, *Proc. 2009 International Workshop on Smart Info-Media Systems in Asia*, pp.73–77 (2009).
17) Andrey, P. and Tarroux, P.: Unsupervised Segmentation of Markov Random Field Modeled Textured Images Using Selectionist Relaxation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.20, No.3, pp.252–262 (1998).
18) Goldberg, D.E.: *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley (1989).
19) Ikeda, K., and Kobayashi, S.: Deterministic Multi-step Crossover Fusion: A Handy Crossover for GAs, *PPSN VII*, pp.162–171 (2002).
20) Hanada, Y., Hiroyasu, T. and Miki, M.: Genetic Multi-step Search in Interpolation and Extrapolation domain, *Proc. GECCO 2007*, pp.1242–1249 (2007).
21) Hanada, Y., Hiroyasu, T. and Muneyasu, M.: Analysis of the Performance of Genetic Multi-Step Search in Interpolation and Extrapolation Domain, *Transactions of the Japanese Society for Artificial Intelligence*, Vol.24, No.1, pp.136–146 (2009) (in Japanese) / *GECCO 2008*, pp.1107–1108 (2008).
22) Asano A., Ohkubo T., Muneyasu, M. and Hinamoto, T.: Primitive and Point Configuration Texture Model and Primitive Estimation using Mathematical Morphology, *Proc. Scandinavian Conference on Image Analysis 2003*, pp.178–185 (2003).
23) Kauffman, S.A.: *Adaptation on Rugged Fitness Landscapes, Lectures in the Sciences of Complexity*, Vol.1, Addison Wesley (1989).
24) Yamada, T. and Nakano, R.: Scheduling by Genetic Local Search with Multi-Step Crossover, *PPSN IV*, pp.960–969 (1996). Genetic Algorithms for Job-Shop Scheduling Problems, *Modern Heuristic for Decision Support*, pp.67–81 (1997).
25) Freisleben, B. and Merz, P.: New Genetic Local Search Operators for the Traveling Salesman Problem, *PPSN IV*, pp.890–899 (1996).
26) Nagata, Y.: New EAX Crossover for Large TSP Instances, *Proc. Parallel Problem Solving from Nature*, *PPSN IX*, pp.372–381 (2006).
27) Fujiki, A., Asano, A. and Muneyasu, M.: Unsupervised Optimization of Mor-

phological Filters for Noise Removal in Texture Images. *Proc. SCIS & ISIS2006*, pp.1794–1799 (2006).
28) Brodatz, P.: *Textures: A Photographic Album for Artists and Designers*, Dover Pubns (1999). http://www.ux.uis.no/˜tranden/brodatz.html
29) Weinberger, E.D.: Correlated and Uncorrelated Fitness Landscapes and How to Tell the Difference, *Biological Cybernetics*, Vol.63, pp.325–336 (1990).
30) Stadler, P.F. and Schnabl, W.: The Landscape of the Travelling Salesman Problem, *Physics Letters A*, Vol.161, pp.337–344 (1992).
31) Merz, P.: Advanced Fitness Landscape Analysis and the Performance of Memetic Algorithms, *Evolutionary Computation*, Vol.12, No.3, pp.303–325 (2004).
32) Fontana, W., Stadler, P.F., Tarazona, P., Weinberger, E.D. and Schuster, P.: RNA Folding and Combinatory Landscapes, *Phys. Rev. E*, Vol.47, No.3, pp.2083–2099 (1993).
33) Markon, S., Arnold, D. V., Bäck, T., Beielstein, T. and Beyer, H.: Thresholding — a Selection Operator for Noisy ES, *Congress on Evolutionary Computation*, pp.465–472 (2001).
34) Beyer, H.: *The Theory of Evolution Strategies*, Springer (2001).
35) Thierens, D. and Goldberg, D.E.: Elitist Recombination: an Integrated Selection Recombination GA. *Proc. 1st IEEE Conference on Evolutionary Computation*, pp.508–512 (1994).
36) Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P.: Optimization by Simulated Annealing, *Science*, Vol.220, No.4598, pp.671–680 (1983).

(Received April 22, 2010)
(Revised June 19, 2010)
(Accepted July 10, 2010)



Yoshiko Hanada received her B.E., M.E. and Ph.D. (Engineering) from Doshisha University, Japan in 2002, 2004 and 2007, respectively. She has been an Assistant Professor of Faculty of Engineering Science, Kansai University, Japan, since 2008. She used to be a JSPS Research Fellow since 2006. Her research interests include the development of novel evolutionary algorithms, image processing and parallel computing. She is a member of IEEE, IPSJ, JSAI, IEICE and JSEC.

Mitsuji Muneyasu received his B.E. and M.E. degrees in system engineering from Kobe University, in 1982 and 1984, respectively and Dr. E. degree from Hiroshima University, Japan in 1993. In 1984, he joined Oki Electric Industry Co., Ltd., in Tokyo, Japan. From 1990 to 1991, he was a Research Assistant at the Faculty of Engineering, Tottori University, Tottori, Japan. From 1991 to 2001, he was a Research Assistant and Associate Professor at the Faculty of Engineering, Hiroshima University, Higashi-Hiroshima, Japan. Since 2001 he joined the Faculty of Engineering, Kansai University, Osaka, Japan, where he is currently a Professor. His research interests include image processing theory and nonlinear digital signal processing. He is a member of IEEE and IPSJ.

Akira Asano is a Professor of the Department of Information Engineering, Graduate School of Engineering, Hiroshima University, Japan. He became a research associate of the Department of Mechanical System Engineering, Kyushu Institute of Technology, Japan, in 1992. He moved to Hiroshima University in 1998, and was promoted to a Professor in 2005. His current research interests are in the areas of mathematical morphology, medical image analysis, visual kansei science, and applied statistics. He is an Associate Editor of the IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences, and a member of Japan Society of Kansei Engineering, Japanese Society of Applied Statistics, and IEEE.