IEICE TRANS. INF. & SYST., VOL.E93–D, NO.9 SEPTEMBER 2010

PAPER

Multi-Objective Genetic Programming with Redundancy-Regulations for Automatic Construction of Image Feature Extractors**

Ukrit WATCHAREERUETAI[†*a)], ***Nonmember***, **Tetsuya MATSUMOTO**[†b)], **Yoshinori TAKEUCHI**[†c)], **Hiroaki KUDO**[†d)], ***and*** **Noboru OHNISHI**[†e)], ***Members***

SUMMARY We propose a new multi-objective genetic programming (MOGP) for automatic construction of image feature extraction programs (FEPs). The proposed method was originated from a well known multi-objective evolutionary algorithm (MOEA), i.e., NSGA-II. The key differences are that redundancy-regulation mechanisms are applied in three main processes of the MOGP, i.e., population truncation, sampling, and offspring generation, to improve population diversity as well as convergence rate. Experimental results indicate that the proposed MOGP-based FEP construction system outperforms the two conventional MOEAs (i.e., NSGA-II and SPEA2) for a test problem. Moreover, we compared the programs constructed by the proposed MOGP with four human-designed object recognition programs. The results show that the constructed programs are better than two human-designed methods and are comparable with the other two human-designed methods for the test problem.

key words: *multi-objective optimization, genetic programming, redundancy regulation, image feature extraction, non-dominated sorting*

1. Introduction

Designing an object recognition program is not an easy task. It is time-consuming and needs much expertise. Usually, object recognition programs are designed by human experts. They generally design an object recognition program based on their knowledge and experience, and sometimes under time-limitations. This implies that they cannot consider all possible programs but only a part of search space can be explored. Therefore, unconventional but potential programs might be ignored and are not discovered.

To cope with difficulty in program designing, many researchers have attempted to create automatic systems for constructing object recognition programs for a given problem. Many approaches exploit evolutionary computation [21], [51] techniques, especially genetic programming (GP) [13], [33], to search for the optimal programs. Some research efforts have attempted to construct a part of object recognition programs, e.g., feature extractor [19], [20], [25], [42], [48], [50], edge detector [49] or interest point detector [22], [23], [26], while some researchers focus on construction of complete object recognition programs [5], [28], [34], [35], [38]. Also various image processing tasks have been studied, e.g., classification [19], [20], segmentation [31], [32], [35], [44], image retrieval [2], or texture analysis [4]. In this work, we focus on automatic construction of feature extraction programs (FEPs) for an image segmentation task. Our approach considered here can automatically construct FEPs without domain-specific knowledge.

In most approaches, including our previous works, single-objective GPs were adopted to optimize performance of constructed programs. However, in practice, we may want to optimize various objectives simultaneously, e.g., to find a program that achieves both high true-positive and high true-negative rates, and prefer various alternatives for decision making. Indeed, there are many works considering multiple objectives but just by integrating multiple objectives into a sole objective function (as in [22], [26], [31], [32], [48]) but this approach cannot identify all of the best solutions (*non-dominated* solutions). Instead, we consider a more sophisticated approach, i.e., to use *evolutionary multi-objective optimization* (EMO) [1], [15], as in [18], [23], [46], [49], [50]. However, we mainly focus on a problem of redundancies in EMO, which is different from the related works.

Nowadays, Pareto-based EMO, such as strength Pareto evolutionary algorithm 2 (SPEA2) [7] or non-dominated sorting genetic algorithm 2 (NSGA-II) [16], have been exploited to solve many problems successfully, especially in genetic algorithm (GA) domain. We tried to adopt the NSGA-II technique in our GP-based FEP construction, so we call it non-dominated sorting GP (NSGP). However, it appears that the NSGP could not work well with our automatic FEP construction system. The main reason seems to be the high redundancies in GP representations. This quite contrasts with GA representations, which usually contain no or low redundancies. The redundancies in GP representation

Manuscript received November 30, 2009.
Manuscript revised April 14, 2010.

[†]The authors are with the Department of Media Science, Graduate School of Information Science, Nagoya University, Nagoya-shi, 464–8601 Japan.

[*]Presently, with International College, King Mongkut's Institute of Technology Ladkrabang, Chalongkrung Road, Ladkrabang, Bangkok 10520 Thailand.

[**]Based on "Construction of image feature extractors based on multi-objective genetic programming with redundancy regulations", by U. Watchareeruetai et al. which appeared in Proceedings of 2009 IEEE International Conference on Systems, Man, and Cybernetics. ©2009 IEEE

a) E-mail: ukrit@ieee.org
b) E-mail: matumoto@is.nagoya-u.ac.jp
c) E-mail: takeuchi@is.nagoya-u.ac.jp
d) E-mail: kudo@is.nagoya-u.ac.jp
e) E-mail: ohnishi@is.nagoya-u.ac.jp

DOI: 10.1587/transinf.E93.D.2614

together with *elitist truncation*, a main aspect of NSGA-II, decrease population diversity rapidly in a few generations, and leads to poor performance in FEP construction.

In this paper, we propose a multi-objective GP (MOGP) technique, which was modified from the NSGP, for automatic construction of FEPs. Three redundancy-regulation mechanisms are introduced to improve population diversity as well as convergence rate. The first is named *semi-elitist truncation* that firstly selects the program with better rank but only one program for each objective value in the objective space. The second is a sampling mechanism that equalizes the probability that each point will be selected, named *phenotypic-uniform sampling*. The third redundancy-regulation, i.e., *prohibition of redundant individuals*, is exploited in offspring generation process; it prohibits offspring that represent programs already discovered.

We have compared the proposed MOGP with two conventional EMOs, i.e., NSGP and SPEA2. Experimental results indicate that the proposed method significantly outperforms the original NSGP (based on NSGA-II) and SPEA2 methods. We also demonstrate that the proposed MOGP method can perform even better than a single-objective GP-based approach in solving a single-objective problem. Finally, we compared the programs constructed by the proposed MOGP with human-designed object recognition programs. The results show that the constructed programs are comparable with two compared methods and are better than the other two compared methods for a test problem.

The rest of this paper is organized as follows. Section 2 provides some backgrounds in multi-objective optimization and description of the NSGA-II. Section 3 explains our GP-based system for FEPs construction. Section 4 describes the proposed MOGP techniques. Test problem is described in Sect. 5. Experimental results and discussion are in Sect. 6. Finally, Sect. 7 concludes the paper.

2. Background and Related Work

2.1 Multi-Objective Optimization and Concept of Domination

The multi-objective optimization problem considered here is a maximization problem. It contains several functions to be maximized as follows:

$$\textit{Maximize } \mathbf{f}(x) = (f_1(x), f_2(x), \ldots, f_m(x)), \tag{1}$$

where x is a solution and $f_1(\cdot), f_2(\cdot), \ldots, f_m(\cdot)$ are m objectives to be maximized. We say that a solution x dominates the other solution y (denoted by $x \succ y$) if the following conditions are satisfied:

$$\begin{aligned} &\forall i \in \{1, 2, \ldots, m\},\ f_i(x) \geqslant f_i(y) \ \wedge \\ &\exists j \in \{1, 2, \ldots, m\},\ f_j(x) > f_j(y). \end{aligned} \tag{2}$$

The ideal goal of multi-objective optimization is to find so called Pareto-optimal front [15]—the set of all possible solutions that are not dominated by the other possible solutions. However, Pareto-optimal front is unknown for most problems, and it may be very difficult to achieve all solutions in the fronts. In practice, we may just prefer a non-dominated front that aligns closely to the Pareto-optimal front. Also a wide spread distribution of the solutions in the front is preferred; more uniform distributions provide wide-range choices to be chosen, compared with compact ones.

2.2 Non-dominated Sorting Genetic Algorithm: NSGA-II

Srinivas et al. [29] proposed non-dominated sorting genetic algorithm (NSGA) in 1995, and its improved version called NSGA-II was proposed in 2000 by Deb et al. [16]. NSGA-II was mainly designed to resolve three criticisms of NSGA, i.e., computational complexity, lack of elitism, and the need for specifying a sharing parameter. NSGA-II has been adopted to solve many problems successfully, and becomes a popular EMO method over the years.

Pseudo code in Fig. 1 describes how NSGA-II works. The main concept of NSGA-II is based on *non-dominated sorting* (step 2). As shown in Fig. 2, a set of all solutions that do not dominate each other are grouped together (called a *front* F_i), and a rank is assigned to each solution in the front. All solutions in the first front F_1 are not dominated by any solutions in the other fronts (so called *non-dominated front*). In a front F_i, each solution will be assigned a relative distance measurement called *crowded distance*, which is the

1. $R_t \leftarrow P_t \cup Q_t$ (P_t: parent population, Q_t: offspring population, t: generation index)
2. Apply non-dominated sorting to R_t to obtain non-dominated fronts $F = \{F_1, F_2, ..., F_{n_t}\}$, where n_t is the number of fronts
3. $P_{t+1} \leftarrow \phi$ and $i \leftarrow 1$
4. while $|P_{t+1}| + |F_i| \leq N$
   $P_{t+1} \leftarrow P_{t+1} \cup F_i$
   $i \leftarrow i + 1$
5. Sorting F_i based on crowded distance (in descending order), $j \leftarrow 1$
6. while $|P_{t+1}| < N$
   $P_{t+1} \leftarrow P_{t+1} \cup s_j$, where $s_j \in F_i$
   $j \leftarrow j + 1$
7. Create new offspring Q_{t+1} from P_{t+1}

Fig. 1 Pseudo code of NSGA-II.

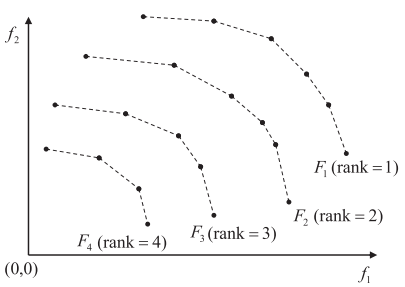


Fig. 2 Example of rank assignment by non-dominate sorting (in the case of two objectives).

average distance (in objective space) from itself to the adjacent solutions.

To create new parent populations, *elitist truncation* (steps 3–6) is adopted. Specifically, the truncation is firstly based on the rank (step 4). It firstly fills the new parent population P_{t+1} by F_1, following by F_2, F_3, and so on. When a front F_j is being considered, the free space of the new parent population might be less than the number of solutions in F_j (so it is the last front to be put into P_{t+1}). Here the truncation changes from rank-based criterion to crowded distance based criterion to preserve diversity of solutions (steps 5–6). The solutions with higher distance will be selected before the lower ones.

The rank and crowded distance are also considered in selection process (step 7), in which binary tournament selection (pool size of two) is performed. The solutions with better rank are selected with the first priority, and if two solutions have the same rank, the solution with higher crowded distance is preferred. The selected solutions will be applied with genetic operators, i.e., crossover or mutation, to generate the new offspring population Q_{t+1} (step 7).

3. Evolutionary Construction of Feature Extraction Programs

3.1 System Overview

Figure 3 is the overview of evolutionary system for construction of FEPs. In this system, inputs needed from users are just image processing library, training images, and objective function(s)[†]. Image processing library consists of basic image processing operations, e.g., edge detection, low-pass filtering, image thresholding. The list of all basic image processing operations is shown in Appendix. These operations are used as primitive operations (POs) for FEP construction. Using the basic image processing operations as POs, instead of simple arithmetic operations (e.g., plus, minus, multiplication) as in [28], [49], [50], would result in an ability to construct more complex image processing programs but with more compact representations.

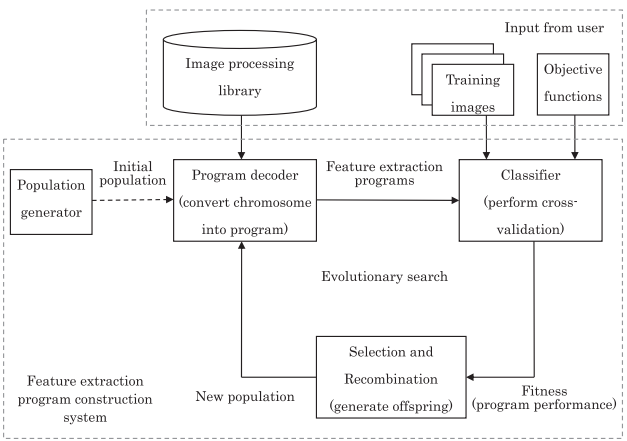


Fig. 3 Overview of an evolutionary system constructing feature extraction programs.

Firstly, the system randomly generates an initial population of solutions (or chromosomes), which encode FEPs. These individuals are then interpreted into FEPs and are evaluated. In the evaluation process, the defined objective function(s) is used to compute fitness, which describes performance of the program. The solutions with the higher performance will have higher chance to survive and be evolved by recombination operators, i.e., crossover and mutation, to generate offspring; whereas the solutions with low performance become extinct. After evolution process finished, the program that gives the best fitness is considered as the output of the system.

3.2 Representation and Decoding

We adopt *linear genetic programming* (LGP) representation [24], instead of tree-based GP (e.g., as in [18], [23], [50]) or graph-based GP (e.g., as in [5], [35]), because of its advantages over the other representations: 1) linear representation is simple but powerful enough to represent graph-based programs, which are more general than tree-based programs, 2) its representation is more compact than tree-based programs [24], 3) it is easy to adopt with various data types, e.g., numerical and image data[††], and 4) there is an efficient algorithm to remove *ineffective codes* (*structural introns*) from representations [24], resulting in reduction of wasteful computation.

In LGP, a program is represented as a sequence of instructions (fixed- or variable-length), and program execution is based on a set of shared registers. In particular, each instruction is encoded by operation code, which indicates PO to be executed, and arguments that specify input and output registers. Program execution starts from the first operation in the sequence, and move to the next operation sequentially. Because image processing operations are utilized as POs, a special register type, i.e., image register (R_I), is needed to store input and processed images. We also need some numerical registers (R_N) to store some parameters of POs.

Our LGP representation is slightly different from the original representation; we use sub-program structure as shown in Fig. 4. One linear program consists of multiple sub-programs; each is executed independently of the others. Each sub-program generates one feature image, which is the content stored in a pre-defined image register after sub-program execution finished. Once all sub-programs are executed, feature images are inputted into a classifier to obtain recognition result. We use Bayesian classifier with histogram approximation [36], in which a pattern distribution

[†]There are other parameters needed to be specified, such as population size, crossover rate, mutation rate, or even termination criterion. However, for *novice users* who do not familiar with evolutionary computation, the default values provided by a *system developer* should be adopted. For *advanced users* who understand the effect of these parameters, they would be allowed to specify these parameters themselves.

[††]In graph-based GP, we may need some constraints to avoiding mis-matching of input-output data.

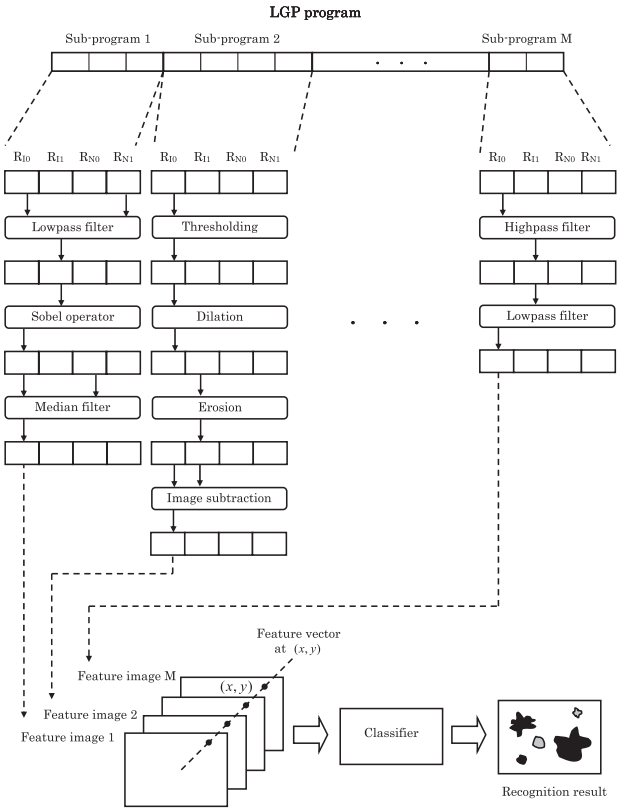


Fig. 4 Example of sub-program structure and its execution.

is approximated by using a histogram, as the classifier.

3.3 Genetic Operators

Genetic operators, i.e., crossover and mutation, are applied to selected solutions (called parents) to produce new solutions (called offspring). Crossover we used is parameterized uniform crossover [47] with 0.2 probability of instruction exchange between two parents. Also a crossover operator that allows swapping of entire sub-programs is adopted. Probability that each crossover type will be used is equal. Mutation operator used here randomly inserts, deletes, or modifies an instruction.

3.4 Program Evaluation

In our previous work [44], a single-objective approach, we used recognition accuracy as fitness value. In the case that the numbers of object and background pixels are not in balance, e.g., objects are relatively small, GP tried to optimize recognition accuracy by avoiding false positive, resulting in missing inner-boundary of objects and even missing entire small objects.

In this work, we add the other performance measure, i.e., true positive rate, to force GP find more object pixels. In evaluation process, *leave-one-out cross validation* [36] is adopted. Specifically, validation is done T times, where T is the number of training images. In each time, $T-1$ images are used for classifier training and the remaining one image is for validation. From all validations, we find the numbers of true positive (tp), true negative (tn), false positive (fp), and false negative (fn) pixels. Then recognition accuracy ACC and true positive rate TP are computed as shown in Eqs. (3) and (4), respectively.

$$ACC = \frac{tp + tn}{tp + tn + fp + fn} \tag{3}$$

$$TP = \frac{tp}{tp + fn} \tag{4}$$

3.5 Redundancy and Canonical Transformation

Generally, GP representations, including tree-, linear-, or graph-based GPs, contains a lot of redundancies, i.e., a program may be represented by different chromosomes. There are several causes of redundancies in GP representations. Although *introns* [14], i.e., ineffective instructions that have no effect on program output, have been shown that they have an advantage in reduction of destructive crossover [14], [30], the existence of introns is one of the main causes of redundancy in GP representations. The other causes are protection mechanism, program structure itself, and so on.

In [44], we have identified the causes of redundancies in LGP-based representations, and proposed a *canonical transformation* for converting original LGP representation into *canonical form* in which structural redundancies[†] are removed. In canonical form, it is easy to verify whether two chromosomes represent the identical program. The use of the canonical transformation allows GP representation to contain introns (to help reduce destructive crossover), and also enables GP to identify whether the considering chromosomes are redundant or not. It can be adopted to improve GP with various techniques [10], [44], [45].

4. Proposed Method

4.1 Problems and Motivation

In this paper, we attempt to implement a MOGP based on the concept of NSGA-II, and we call it non-dominated sorting GP (NSGP). However, the NSGP could not work well in our evolutionary FEP construction system. The reasons would be related with a difference in characteristics of GA and GP problems—redundancy level. In many GA problems, chromosome representations do not contain any redundancy—genotype-phenotype mapping is one-to-one. On the contrary, chromosome representations in GPs often have high-level redundancies (e.g., as described in Sect. 3.5). This means that the mapping from the genotype

[†]In the canonical transformation, we detect and remove only *structural introns* [24]. *Semantic introns* [24] are not considered because they strongly depend on primitive operators used and there is no general and efficient method to detect them. Although there is an algorithm proposed to detect semantic introns [24] but it is not practical (need a huge number of program evaluations).

1. $R_t \leftarrow P_t \cup Q_t$ (P_t: parent population, Q_t: offspring population, t: generation index)
2. Apply non-dominated sorting to R_t to obtain non-dominated fronts $F = \{F_1, F_2, ..., F_{n_t}\}$, where n_t is the number of fronts
3. Subdivide each F_i into groups G_{ij} that contains the solutions with same objective values ($F_i = \bigcup_j^{m_i} G_{ij}$, where m_i is the number of groups).
4. $P_{t+1} \leftarrow \phi$, $i \leftarrow 1$, and $j \leftarrow 1$
5. while $|P_{t+1}| \leq N$

   if $i = n_t$ then $i \leftarrow 1$
   if $j = m_i$ then $j \leftarrow 1$ and $i \leftarrow i + 1$
   if $|G_{ij}| > 0$ then randomly select a solution s from G_{ij}, $G_{ij} \leftarrow G_{ij} - \{s\}$, and $P_{t+1} \leftarrow P_{t+1} \cup \{s\}$
   $j \leftarrow j + 1$

6. Calculate sampling probability and averaged crowded distance for each solution in P_{t+1}
7. Create new offspring Q_{t+1} from P_{t+1} by using phenotypic-uniform sampling, big tournament size, and redundant-offspring prohibition

Fig. 5 Pseudo code of the proposed MOGP.

(representation) space into phenotype (program) space is many-to-one. Moreover, the mapping from phenotype space into objective space is generally many-to-one too, i.e., it is possible that two completely different programs may lead to the same objective values. Consequently, elitism truncation in NSGA-II seems to make population lose diversities rapidly. In addition, it seems that the difference in structure difficulties appears, like in tree-based GP [12]. In other words, some programs may be produced easily, whereas the other programs are rarely produced. A solution in the non-dominated front that is easily produced may occupy the most population space within a few generations, resulting in diversity loss. These motivated us to develop the ways to regulate redundancies in MOGP-based FEP construction.

4.2 Non-dominated Sorting with Redundancy-Regulations

We propose a MOGP improved from NSGA-II based GP for automatic construction of FEPs. The proposed method was designed to resolve the redundancy-related problems described in the previous section. It is described as the pseudo code in Fig. 5. The key different points of the proposed method are listed as follows:

- *Semi-elitist truncation (steps 3–5)*: Similar to the elitist truncation, it firstly fills the new parent population P_{t+1} with solutions in F_1, following by F_2, F_3, and so on. However, for each point (objective value) in the objective space, it randomly choose only one solution $s \in G_{ij}$. If all fronts have been considered but P_{t+1} is still not full, this process will be repeated. Note that the selected programs cannot be selected again; they are removed from G_{ij}. By doing that, we can maintain diversity of population while guarantee that the each elite point still exists in the population (if the number of the elite points are not greater than the population size).
- *Phenotypic-uniform sampling (in steps 6–7)*: Instead of using uniform sampling to choose solutions to be competed in tournament selection, we use a sampling mechanism that is likely to select the solution that solely locates at a point or there are less other solutions located at the same point. Let's assume that the considering population P_{t+1} corresponds to N_{diff} different points (objective values) in the objective space. The probability that a solution s locates at a point p_i ($i \leq N_{diff}$) will be selected is defined as $1/(N_{diff} * C_i)$, where C_i is the number of solutions that are located in p_i. It means that each point has equal probability to be selected.
- *Prohibition of redundant individuals (in step 7)*: Once an offspring is produced, we transform it into canonical form [44] and verify whether it represents a program discovered before in the evolutionary search. If it is a redundant program (already discovered), we apply mutation operator on that individual until it becomes the new one that has not been discovered before. In [44], we have demonstrated that the use of such constraint can significantly improve search performance of single-objective GP.
- *Averaged crowding-distance (in steps 6–7)*: The direct use of crowded-distance assignment algorithm in [16] may result in assigning different distance for the solutions located in the same point (some may have zero distance whereas the others have positive distance). Therefore, we find the summation of distance values of all solutions located in the same point (except the extrema), and assign the average value for each point instead.
- *Big tournament size (in step 7)*: The balance between exploration (global search) and exploitation (local search) powers is a crucial issue in evolutionary computation [21]. The above redundancy-regulation mechanisms greatly increase diversity of population, i.e., exploration power is increased. However, there are only a little copies of each elite solutions (non-dominated solutions) existing in each generation. If we use small tournament size (e.g., two), only a few numbers of elite solutions will be exploited in each generation (i.e., very low exploitation power), and this leads to slow convergence. Therefore, big tournament size (in this paper, 10 for population size of 50) is preferred to maintain well balance between exploration and exploitation.

5. Test Problem

In this work, we consider an image segmentation problem, i.e., lawn weed detection. Until now, there are a number of methods that have been proposed for detecting weed in lawn fields [3], [17], [37], [39]–[41], [43]. Lawn weed detection problem considered here is a two-class segmentation problem. The goal is to segment the area of weeds from lawn

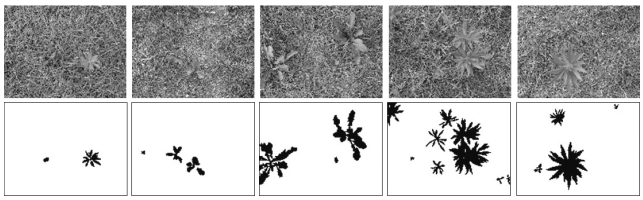


Fig. 6 Lawn weed images (top) and their corresponding ground truths (bottom).

background to perform precision spraying by an automatic weed control system. In particular, the automatic weed control system uses a camera to capture lawn images, and detects weeds existing in the capture images. The system then controls the nozzle system so that it sprays herbicide onto only the area of detected weeds, instead of spraying onto the entire area. This system can help reduce herbicide usage, resulting in cost reduction and safe environment.

Lawn weed images used in this work are divided into two datasets. The first dataset consists of five images. It was used to train the evolutionary system for constructing FEPs. The second dataset consists of 25 images (only 20 images contain weeds while the remaining five images contain no weeds). It was used as a validation set for the last experiment (Sect. 6.3). Note that resized images (160×120 pixels) were used in all experiments, except the last experiment in Sect. 6.3 that used full size images, i.e., 640×480 pixels. Figure 6 shows the lawn weed images in the first dataset and their corresponding ground truths.

6. Experiments and Discussions

To evaluate the performance of the proposed MOGP method, we have conducted three experiments. The first experiment compared FEP construction performance of the proposed methods with two conventional EMOs. In the second experiment, the proposed MOGP-based approach was compared with our previous single-objective approach for a single-objective problem. In the last experiment, we compared the programs constructed by the proposed MOGP with human-designed lawn weed detection methods.

6.1 The Proposed MOGP vs. Conventional EMOs

In this experiment, we compare performance of the proposed MOGP with two conventional EMOs, i.e., NSGA-II (called NSGP) and SPEA2 [7], in FEP construction. Two objective functions described in Eqs. (3) and (4), i.e., recognition accuracy ACC and true positive rate TP, were used. Their parameters were set as shown in Table 1. The NSGP, SPEA2, and proposed methods were executed 30 times, and their average results were compared.

6.1.1 Comparison of Hypervolume and Coverage

Performance comparison was done based on two complementary measures used in [6]. The first measure is *hypervolume*—the summation of block areas under the non-dominated front (Fig. 7). An algorithm that provides large hypervolume implies that it found non-dominated front with well distribution and/or good convergence. The second measure is *coverage* $C(A \succ B)$, which is the fraction of non-dominated solutions found by an algorithm B that are dominated by at least one solution found by an algorithm A. It is defined as Eq. (5):

$$C(A \succ B) = \frac{|S_{A \succ B}|}{|S_B|}, \tag{5}$$

$$S_{A \succ B} = \{y \in S_B | x \succ y, \exists x \in S_A\}, \tag{6}$$

where S_A and S_B are the sets of non-dominated solutions found by the algorithms A and B, respectively, and $|\cdot|$ denotes the cardinality of a set. Note that $C(A \succ B)$ is not usually equal to $1 - C(B \succ A)$; consequently, both $C(A \succ B)$ and $C(B \succ A)$ need to be considered together. The coverage measure provides relative information on convergence between two algorithms.

The comparison based on the hypervolume measure is shown in Table 2. As we expected, the proposed method provides larger hypervolume than the NSGP and SPEA2 methods because redundancy-regulation mechanisms would guide GP to find non-dominated solutions with (at least)

Table 1 List of parameters.

Parameter	Setting
Population size	50
Max. generations	50
Crossover operator	parameterized uniform [47] and sub-program crossovers
Mutation operator	insertion, deletion, modification
# crossover offspring	24 (48%)
# mutated offspring	26 (52%)
Tournament size	NSGP: 2 SPEA2: 2 proposed: 10
# sub-programs	2
Max. sub-program length	20 operations
# image registers	4
# numerical registers	4
# primitive operations	51

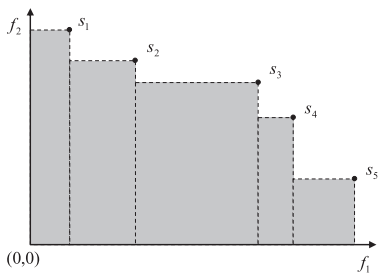


Fig. 7 Hypervolume of two-objective case (origin is the reference point). s_i stands for solution i in the non-dominated front.

better distribution. The t-test confirms these two distributions (NSGP vs. the proposed MOGP and SPEA2 vs. the proposed MOGP) are significantly different. Moreover, we compare the proposed MOGP with NSGP and SPEA2 in each trials, and count the number of trials in which the considering method is better. From Table 2, the proposed MOGP is better than the NSGP for 27 trials and always better than the SPEA2 in this hypervolume measure.

Table 2 Comparison of hypervolume.

	NSGP	**SPEA2**	**Proposed**
Average	0.5543	0.4966	0.6843
STD	0.0854	0.0881	0.0470
t-test [a]	3.29×10^{-7}	1.09×10^{-9}	N/A
# of better trials [a]	3	0	27, 30

[a] NSGP or SPEA2 vs. the proposed MOGP

Table 3 Comparison of coverages.

Proposed vs. NSGP (A: proposed, B: NSGP)	
$C(A \succ B)$	0.9196
$C(B \succ A)$	0.0394
# trials in which $C(A \succ B) > C(B \succ A)$	29
Proposed vs. SPEA2 (A: proposed, B: SPEA2)	
$C(A \succ B)$	0.9889
$C(B \succ A)$	0.0167
# trials in which $C(A \succ B) > C(B \succ A)$	30

To assure its superiority in convergence, we have to consider the coverage measure shown in Table 3. From the result, most solutions found by the NSGP and SPEA2 methods are dominated by those of the proposed method. Also the proposed method is better in the coverage measure for 29 and 30 trials, comparing with the NSGP and SPEA2 respectively. It obviously suggests that the proposed method provides much better convergence than the NSGP and SPEA2 methods. Example results of the first nine trials shown in Fig. 8 ensure that the proposed method outperforms the two conventional EMOs in both convergence rate and distribution diversity.

6.1.2 Comparison of the Non-dominated Front Size

Table 4 shows the average size of the non-dominated front and the average number of different points in the front. In the case of the NSGP and SPEA2 methods, the non-dominated front size often reached population size but most solutions in the front were redundant. The result of a trial shown in Fig. 9 demonstrates that redundant solutions do not distribute uniformly (see the nearby numbers in Fig. 9). Instead, population seems to converge to a solution that is easy

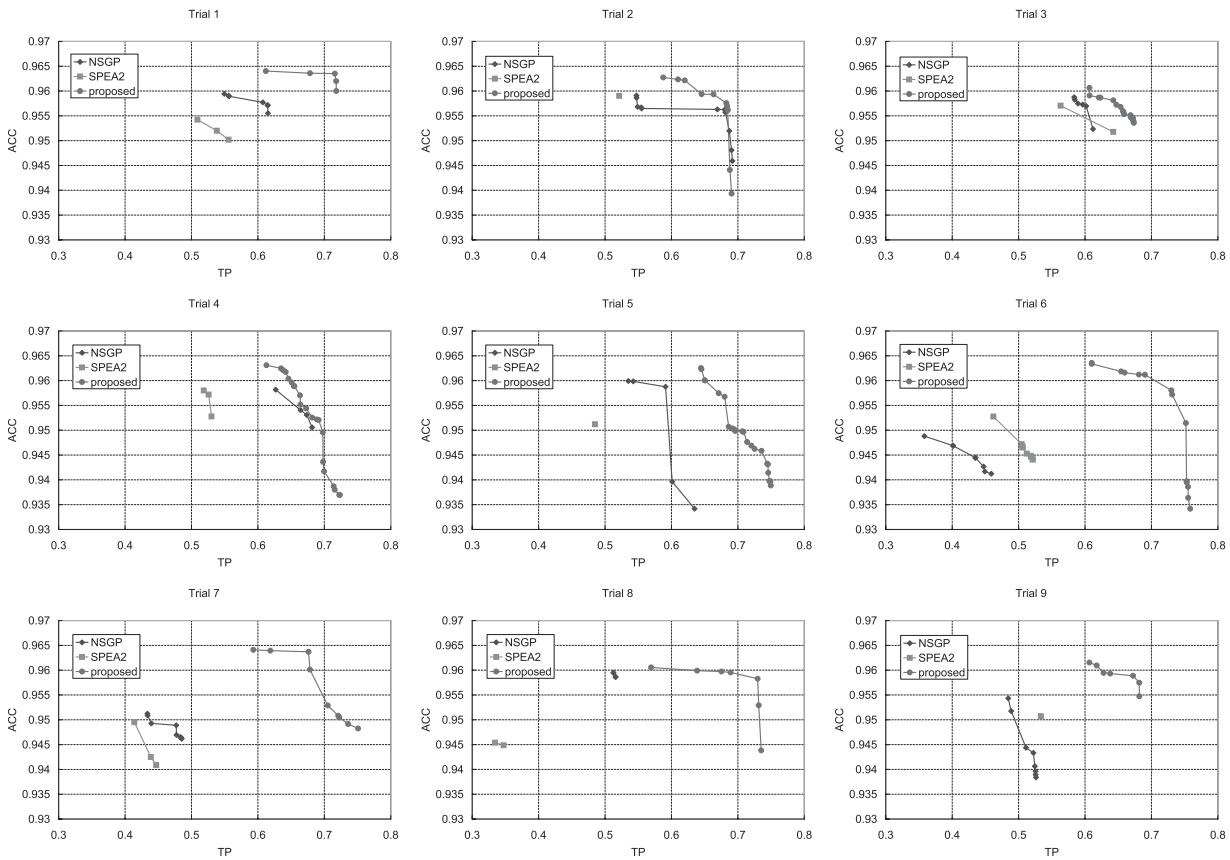


Fig. 8 Comparison of the non-dominated fronts obtained by the NSGP, SPEA2, and proposed methods (the first nine trials).

Table 4 Comparison of non-dominated front size.

	NSGP	SPEA2	Proposed
Average 1^{st} front size	45.87	44.40	14.23
Average no. of different points	5.60	3.53	13.63
Ratio	0.14	0.08	0.96

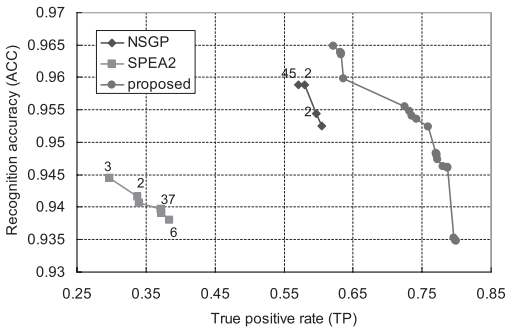


Fig. 9 Example of the non-dominated fronts obtained from the NSGP, SPEA2, and proposed methods. The numbers in the graph indicate the number of copies of points (if no number, there is only one copy).

to be produced (different structure difficulties [12]). These reveal an adverse effect of the elitist truncation in high-redundancy EMO. In the case of the proposed method, the ratio between these two values is nearly one, i.e., most copy points were removed. This would be mainly caused by the use of semi-elitist truncation. Also the average size of non-dominated front is far smaller than the population size. This implies that many fronts were preserved and rank-based selection would still properly function.

6.2 Single-Objective vs. Multi-Objective

In the previous section, we have demonstrated that the proposed MOGP method is better than the NSGP and SPEA2 methods in FEP construction. Here we compare MOGP-based approaches (the proposed MOGP, NSGP, and SPEA2) with a single-objective based approach for solving a single-objective problem. Comparison was done based on one objective, i.e., recognition accuracy ACC (Eq. (3)). The single-objective GP with prohibition of producing discovered-offspring described in [44] was experimented. The same parameters as in the previous section were used. The average recognition accuracy (over 30 independent trials) of the best individual from each trial is shown in Table 5.

The results indicate that the NSGP and SPEA2 could not beat the single-objective approach in this problem; they are better than the single-objective approach for only four and one trials, respectively. On the contrary, the proposed MOGP-based approach outperforms the single-objective approach; it beats the single-objective approach for 21 trials. Again the t-test indicates that these two distributions are significantly different. This result suggests that the use of redundancy-regulations in the proposed MOGP could encourage automatic FEP construction system in solving even a single-objective problem.

Table 5 Comparison of recognition accuracy (ACC) between single-objective and multi-objective approaches.

	Single-objective	Multi-objective		
		NSGP	SPEA2	Proposed
Average	0.9608	0.9567	0.9525	0.9623
STD	0.0034	0.0040	0.0050	0.0015
t-test [a]	N/A	5.55×10^{-4}	1.21×10^{-8}	0.0403
# better trials [a]	26, 29, 9	4	1	21

[a] Single-objective vs. multiobjective

6.3 Automatically Constructed Programs vs. Human-Design Programs

Until now, we have shown that the proposed MOGP with redundancy-regulation mechanisms is better than the compared MOGPs and also the single-objective approach. Herein, we will compare the performances of programs that are automatically constructed by the proposed MOGP with the human-designed programs to observe whether the proposed MOGP can construct human-competitive programs or not. The problem we consider is still the lawn weed detection problems but we evaluate lawn weed detection programs with different measures, i.e., weed control performances. Two main performance measures of automatic weed control system are the performance in weed destruction—how weeds can be destroyed (killed), and spraying error—how large area containing no weeds is sprayed. According to the simulated weed control system in [43], after weed detection, the system divides the detected weed image into small blocks of size 30×60 pixels (therefore, one image contains 16×40 blocks), and it sprays herbicide onto only the blocks containing detected weeds. If a weed is sprayed, they will be counted as a killed (or destroyed) weed. We used the number of killed weeds N_{kw} and the number of false-spray blocks N_{fsb} (in fact we maximize $-N_{fsb}$) as the objectives to be optimized, and constructed lawn weed detection programs by using the proposed MOGP.

Four human-designed lawn weed detection methods are considered here. The first is called Bayesian classifier based method (BC) [43]. The second is morphological operation based method (MO) [43]. The third is gray-scale uniformity analysis method (UA) [39]. The last method is modified from the BC method but adopts support vector machine (SVM) as the classifier, so it is called SVM method. Parameters of these methods (e.g., window size, threshold values) are adjusted and their performance measures are plotted in Fig. 10.

The proposed MOGP have been executed 30 trials. The parameters were set as similar to the previous experiments, except the maximum generation was increased to 500. From these 30 trials, it generated many non-dominated

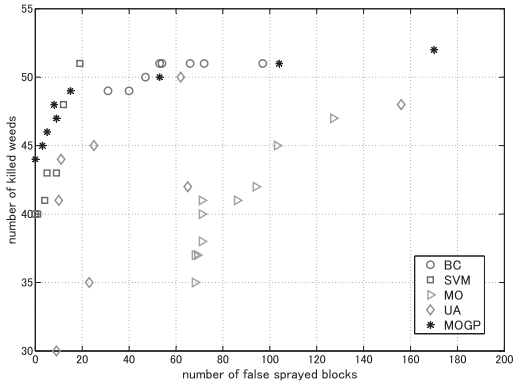


Fig. 10 Comparison of weed control performances, i.e., weed destruction and spraying error. The total number of weeds existing in the dataset is 58.

solutions. To compare with the conventional lawn weed detection methods, we find the solutions that give the number of killed weeds N_{kw} in range of 44–54 and give minimum false-spray blocks N_{fsb}. Their performances are also plotted in Fig. 10. From the figure, we found that the programs constructed by the proposed MOGP outperform two lawn weed detection methods designed by human, i.e., the UA and MO methods, and they are comparable with the other two methods, i.e. the BC and SVM methods. Note that the BC and SVM methods are classifier-based weed detection methods that segment weeds from lawn backgrounds based on two image features, hence it is fair to compare them with the programs constructed by the proposed MOGP (which are classifier-based methods and use two features also).

6.4 Discussions

In Sect. 6.1, we have shown that the proposed MOGP outperforms the NSGP and SPEA2 in both hypervolume and coverage measures. It should be caused by the redundancy-regulation mechanisms, i.e., the semi-elitist truncation, phenotypic-uniform sampling, and prohibition of redundant offspring, that guide the MOGP to search in wider areas.

The use of the semi-elitist truncation allows more genetic materials to be exploited in the recombination process, comparing with the elitist truncation in the NSGP. Moreover, it allows newly born solutions (that often still have low fitness) of the potential areas to be improved later. This is similar to an evolutionary algorithm model called hierarchical fair competition (HFC) [11]. In the HFC model, newly born solutions are protected from competition with high-fitness solutions (be considered as unfair competition) by using sub-population and import-export mechanism. Although in our case, we do not directly protect unfair competitions, we preserve solutions with various fitness values from strong to weak ones. This allows newly born solutions to exist in the population and have a chance to win in a selection and be evolved further. Once population evolves and new better solutions are found, the level of weak solutions will be gradually increased, i.e., threshold level is adapted automatically.

The phenotypic-uniform sampling reduce the effect of different structural difficulties. As shown in Fig. 9, in the cases of the NSGP and SPEA2, the non-dominated fronts are dominated by a few solutions that might easily be produced. Because there is higher number of copies of these solutions, if the conventional uniform sampling is adopted, these solutions will have more chance to be selected and evolved further, and might produce more copies, resulting in diversity loss. However, we did not see this effect in the proposed MOGP, which adopts the phenotypic-uniform sampling. This is because the phenotypic-uniform sampling equalizes the probability that each point (in the objective space) will be selected.

The prohibition of redundant offspring is also very important mechanism. Although the use of semi-elitist truncation and phenotypic-uniform sampling increases the varieties of the solutions to be evolved, they do not guarantee that the generated offspring are the new programs that have not been discovered yet. On the contrary, the use of prohibition of redundant offspring forces the MOGP to find out a new program whenever it generates an offspring. This helps accelerate evolutionary process so much. It is the main reason that the proposed MOGP outperforms the NSGP and SPEA2 in the coverage measure.

In Sect. 6.2, we have shown that the proposed MOGP outperforms the single-objective approach in solving the single-objective problem. Because both the single-objective approach and the proposed MOGP exploit the prohibition of redundant offspring; consequently, the main reason of this success should be the use of the semi-elitist truncation and phenotypic-uniform sampling. Although the single-objective approach can find out new programs in each generation (by using repeated mutation), the evolutionary search often starts from only one point, i.e., the solution with best fitness in that time. However, there are many points to start the search in the case of the proposed MOGP (the solutions in the non-dominated fronts), hence it can search for the new programs in wider area, resulting in better performance.

Finally, in Sect. 6.3, we have shown that the programs constructed by the proposed MOGP are better than the compared lawn weed detection, i.e., the UA and MO methods, and are comparable with the BC and SVM methods. It demonstrates a success of the FEP construction by the proposed MOGP. The reasons of this success should be that a huge number of solutions have been considered and really tested with the training dataset, and the redundancy-regulation mechanisms guide the MOGP to search in wider areas and to find out new solutions in each generation. That means 25,000 solutions have been generated in each trial in the experiment. Moreover, the proposed MOGP based FEP construction might found unconventional but potential solutions that are ignored by human experts.

Compared with the NSGP, the proposed method requires much computation cost due to the additional mechanisms. Among the three mechanisms, the prohibition of redundant individuals seems to be the most computation-

ally intense process. However, in our previous work, we have measured computation time for the canonical transformation and comparison of canonical forms (which are used in this mechanism), and we have found that it took very short computation time (0.01% of the total computation time), compared with that of fitness evaluation (99.98% of the total computation time). Therefore, the computation cost increased by the additional mechanisms can be disregarded.

In this work, we have conducted experiments for the case of two objectives only. However, the number of objectives might be more than two. For the cases of many objectives (e.g., more than four), the proposed method might not work well because the number of non-dominated solutions will be increase exponentially [9]. In this case, we may need to adopt some techniques such as modification of domination concept [9],ϵ-ranking [8], or substitute distance assignment scheme [27] to improve selection pressure on some solutions in the non-dominated front.

7. Conclusion

We have proposed a MOGP for automatic construction of FEPs. The proposed method was modified from the NSGA-II, by including three redundancy-regulation mechanisms, i.e., semi-elitist truncation, phenotypic-uniform sampling, and prohibition of redundant offspring. We have conducted three main experiments to assess the performance of the proposed MOGP. Lawn weed detection problem was used as the test problem.

First, we compared the proposed MOGP with conventional NSGP and SPEA2 in a task of FEP construction. Experimental results yield that the proposed MOGP outperforms the compared methods in both convergence rate and diversity of solutions. It shows the use of redundancy-regulation mechanisms can effectively help preserve population diversity, resulting in more distinct solutions in the non-dominated front.

Second, we compared the proposed MOGP, NSGP (based on the concept of NSGA-II), and SPEA2 with a single-objective approach to construct FEPs for a single-objective problem. The results indicate that the proposed MOGP can generate programs better than the single-objective approach, whereas the NSGP and SPEA2 cannot. It yields the advantages of redundancy-regulations in encouraging the evolutionary search to find better program even in the single-objective problem.

Finally, we compared the programs automatically constructed by the proposed MOGP with the four human-designed lawn weed detection methods. The results indicate that the constructed programs are better than two compared methods, and are comparable with the other two methods. This demonstrates the success of the proposed MOGP-based system for constructing image feature extractors.

Acknowledgements

This work was supported by a research grant from the Hori Information Science Promotion Foundation.

References

[1] A. Abraham, L. Jain, and R. Goldberg (eds.), Evolutionary Multiobjective Optimization: Theoretical Advances and Application, Springer-Verlag, USA, 2005.
[2] A. Dong, B. Bhanu, and Y. Lin, Evolutionary Feature Synthesis for Image Databases, in Genetic and Evolutionary Computation for Image Processing and Analysis, eds. S. Cagnoni, E. Lutton, and G. Olague, pp.325–344, Hindawi, 2007.
[3] A. Otsuka and K. Taniwaki, "Round leaf weed detection in lawn field using texture analysis," Proc. 55th JSAM Annual Meeting, pp.235–236, 1996.
[4] A. Song and V. Ciesielski, "Texture analysis by genetic programming," Proc. CEC 2004, vol.2, pp.2092–2099, 2004.
[5] A. Teller and M. Veloso, "PADO: A new learning architecture for object recognition," Symbolic Visual Learning, eds. K. Ikeuchi and M. Veloso, pp.77–112, Oxford Univ. Press, Oxford, UK, 1997.
[6] E. Zizler and L. Thile, "Multiobjective optimization using evolutionary algorithms: A comparative study," in Parallel Problem Solving from Nature V, ed. A.E. Eiben, pp.292–301, Amsterdam, 1998.
[7] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," Tech. Rep. 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.
[8] H. Aguirre and K. Tanaka, "ϵ-ranking for effective many objectives optimization on MNK-landscapes," IPSJ Online Trans., vol.2, pp.225–239, 2009.
[9] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," Proc. IEEE CEC-2008, pp.2424–2431, Hong Kong, China, 2008.
[10] J. Niehaus, C. Igel, and W. Banzhaf, "Reducing the number of fitness evaluations in graph genetic programming using a canonical graph indexed database," Evol. Comput., vol.15, no.2, pp.199–221, 2007.
[11] J.J. Hu and E.D. Goodman, "The hierarchical fair competition (HFC) model for parallel evolutionary algorithms," Proc. IEEE Congress on Evolutionary Computation (CEC-02), pp.49–54, 2002.
[12] J.M. Daida, H. Li, R. Tang, and A.M. Hilss, "What makes a problem GP-hard?: Validating a hypothesis of structural causes," Proc. GECCO-2003, LNCS 2724, pp.1665–1677, Springer-Verlag, 2003.
[13] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992.
[14] J.R. Levenick, "Inserting introns improves genetic algorithm success rate: Taking a cue from biology," Proc. ICGA-91, eds. R.K. Belew and L.B. Booker, pp.123–127, 1991.
[15] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, Wiley, 2001.
[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol.6, no.2, pp.182–197, 2002.
[17] K. Kawamura, T. Mashita, Y. Miwa, and A. Ito, "Developing of weeding robot (2): Development of weed detecting sensors on green area of golf course," Proc. JSPE, pp.443–444, 1993.
[18] K. Krawiec, "Generative learning of visual concepts using multiobjective genetic programming," Pattern Recognit. Lett., vol.28, no.16, pp.2385–2400, 2007.
[19] K. Krawiec and B. Bhanu, "Visual learning by coevolutionary feature synthesis," IEEE Trans. Syst. Man Cybern. B, Cybern., vol.35, no.3, pp.409–425, 2005.
[20] K. Krawiec and B. Bhanu, "Visual learning by evolutionary and coevolutionary feature synthesis," IEEE Trans. Evol. Comput., vol.11,

no.5, pp.635–650, 2007.
[21] K.A. De Jong, Evolutionary Computation: A Unified Approach, MIT Press, 2006.
[22] L. Trujillo and G. Olague, "Automated design of image operators that detect interest points," Evol. Comput., vol.16, no.4, pp.483–507, 2008.
[23] L. Trujillo, G. Olague, E. Lutton, and F. Fernández de Vega, "Multi-objective design of operators that detect points of interest in images," Proc. GECCO-08, Atlanta, Georgia, USA, pp.1299–1306, 2008.
[24] M. Brameier and W. Banzhaf, Linear Genetic Programming, Springer, 2007.
[25] M. Ebner, "Evolution of hierarchical translation-invariant feature detectors with an application to character recognition," Proc. DAGM-Symposium 1997, pp.456–463, Braunschweig, Germany, 1997.
[26] M. Ebner and A. Zell, "Evolving a task specific image operator," Proc. First European Workshops on Evolutionary Image Analysis, Signal Processing, and Telecommunications, LNCS 1596, pp.74–89, 1999.
[27] M. Köppen and K. Yoshida, "Substitute distance assignments in NSGA-II for handling many-objective optimization problems," Proc. EMO 2007, LNCS 4403, pp.727–741, 2007.
[28] M. Zhang, V. Ciesielski, and P. Andreae, "A domain-independent window approach to multiclass object detection using genetic programming," EURASIP J. Appl. Signal Process., vol.8, pp.841–859, 2003.
[29] N. Srinivas and K. Deb, "Multiobjective function optimization using nondominated sorting genetic algorithms," Evol. Comput., vol.2, no.3, pp.221–248, 1995.
[30] P. Nordin, F.D. Francone, and W. Banzhaf, "Explicitly defined introns and destructive crossover," Advance in Genetic Programming II, eds. P.J. Angeline and K. Kinear, pp.111–134, MIT Press, 1996.
[31] R. Poli, "Genetic programming for feature detection and image segmentation," Evolutionary Computing, LNCS 1143, pp.110–125, 1996.
[32] R. Poli, "Genetic programming for image analysis," Proc. GECCO-96, pp.363–368, Stanford, CA, 1996.
[33] R. Poli, W.B. Langdon, and N.F. McPhee, A Filed Guide to Genetic Programming, Published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk, 2008.
[34] S. Aoki and T. Nagao, "Automatic construction of tree-structural image transformations using genetic programming," Proc. ICAIP-99, pp.136–141, Venezia, Italy, 1999.
[35] S. Shirakawa and T. Nagao, "Genetic image network (GIN): Automatically construction of image processing," Proc. IWAIT-2007, pp.643–648, Bangkok, Thailand, 2007.
[36] S. Theodoridis and K. Koutroumbas, Pattern Recognition, third ed., Academic Press, 2006.
[37] T. Mashita, A. Ito, and Y. Miwa, "Developing of weeding robot (1): Manufacture of weed discrimination system on golf course," Proc. JSPE, pp.997–998, 1992.
[38] T. Nagao and S. Masunaga, "Automatic construction of image transformation processes using genetic algorithm," Proc. ICIP-96, vol.3, pp.731–734, Lausanne, Switzerland, 1996.
[39] U. Ahmad, N. Kondo, S. Arima, M. Monta, and K. Mohri, "Weed detection in lawn field based on gray-scale uniformity," J. Jpn. Soc. Environ. Control Biol., vol.36, no.4, pp.227–237, 1998.
[40] U. Ahmad, N. Kondo, S. Arima, M. Monta, and K. Mohri, "Weed detection in lawn field using machine vision: Utilization of textural features in segmented area," J. JSAM, vol.61, no.2, pp.61–69, 1999.
[41] U. Ahmad, N. Kondo, S. Arima, M. Monta, and K. Mohri, "Weed center detection in lawn field using morphological image processing," J. SHITA, vol.11, no.2, pp.127–135, 1999.
[42] U. Watchareeruetai, T. Matsumoto, N. Ohnishi, H. Kudo, and Y. Takeuchi, "Acceleration of genetic programming by hierarchical structure learning: A case study on image recognition program synthesis," IEICE Trans. Inf.& Syst., vol.E92-D, no.10, pp.2094–2102, Oct. 2009.
[43] U. Watchareeruetai, Y. Takeuchi, T. Matsumoto, H. Kudo, and N. Ohnishi, "Computer vision based methods for detecting weeds in lawns," Mach. Vis. Appl., vol.17, no.5, pp.287–296, 2006.
[44] U. Watchareeruetai, Y. Takeuchi, T. Matsumoto, H. Kudo, and N. Ohnishi, "Transformation of redundant representations of linear genetic programming into canonical forms for efficient extraction of image features," Proc. IEEE CEC-2008, pp.1996–2003, Hong Kong, China, 2008.
[45] U. Watchareeruetai, Y. Takeuchi, T. Matsumoto, H. Kudo, and N. Ohnishi, "Efficient construction of image feature extraction programs by using linear genetic programming with fitness retrieval and intermediate-result caching," Foundation of Computational Intelligence Volume 4: Bio-inspired Data Mining, eds. A. Abraham, et al., pp.355–375, Springer-Verlag, 2009.
[46] W. Jaskowski, K. Krawiec, and B. Wieloch, "Multitask visual learning using genetic programming," Evol. Comput., vol.16, no.4, pp.439–459, 2008.
[47] W.M. Spears and K.A. De Jong, "On the virtues of parameterized uniform crossover," Proc. ICGA-91, eds. R.K. Belew, L.B. Booker, and Morgan Kaufmann, pp.230–236, 1991.
[48] Y. Lin and B. Bhanu, "Object detection via feature synthesis using MDL-based genetic programming," IEEE Trans. Syst. Man Cybern. B, Cybern., vol.35, no.3, pp.538–547, 2005.
[49] Y. Zhang and P.I. Rockett, "Evolving optimal feature extraction using multi-objective genetic programming: A methodology and preliminary study on edge detection," Proc. GECCO-05, pp.795–802, Washington DC, USA, 2005.
[50] Y. Zhang and P.I. Rockett, "Domain-independent feature extraction for multi-classification using multiobjective genetic programming," Pattern Anal. Appl., DOI 10.1007/s10044-009-0154-1, published online: April 08, 2009.
[51] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Berlin, Germany, 1996.

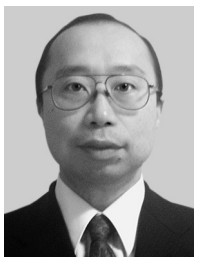
Appendix: Primitive Operations

Table A· 1 shows the list of all basic image processing operations we used in this work.

Table A· 1 Primitive operations used in this work.

One-input operations	Two-input operations
image → image	**image, image → image**
highpass filter	image addition
Sobel operation	image subtraction
image negative	**image, real value → image**
mean thresholding	lowpass filter
entropy thresholding	median filter
histogram equalization	morphological dilation
image → real value	morphological erosion
global mean	morphological opening
global variance	morphological closing
global STD	local histogram equalization
global skewness	thresholding
global kurtosis	local variance
global maximum	local skewness
global minimum	local kurtosis
global median	local maximum (max filter)
global mode	local minimum (min filter)
global range	local mode
global entropy	local range
	local entropy

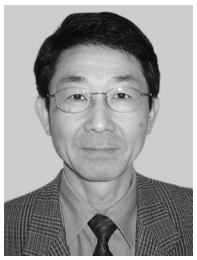
Ukrit Watchareeruetai received the B.Eng. in Electrical Engineering from Kasetsart University (Thailand) in 2002. From 2002–2004, he worked as a research assistant at Kasetsart Signal and Image Processing Laboratory. He received the Master of Information Science and Doctor of Engineering degrees from the Graduate School of Information Science, Nagoya University (Japan) in 2007 and 2010, respectively. Currently, he is a lecturer at International College, King Mongkut's Institute of Technology Ladkrabang. His research interests include computer vision, pattern recognition and evolutionary computation. He is a member of IEEE, ACM and IPSJ.



Tetsuya Matsumoto received the B.E., M.E., and Dr. Eng. degrees from Nagoya University, Nagoya, Japan, in 1982, 1984 and 1996, respectively. From 1984 to 1989, he worked in Toshiba Corporation, Fuchu, Japan, where he was engaged in research and development of the control systems of nuclear power plants. In 1993, he joined the Education Center for Information Processing, Nagoya University. In 1998, he moved to the Department of Information Engineering, Graduate School of Engineering, Nagoya University. His current interests include neural networks, image processing and machine learning. He is a member of JNNS and JSAI.

Yoshinori Takeuchi received the degrees of B. Eng., M. Eng. and Dr. Eng. from Nagoya University in 1994, 1996 and 1999, respectively. In 1999, he was a Research Fellow of the Japan Society for the Promotion of Science. In 2000, he was a member of the Graduate School of Engineering, Nagoya University. Currently, he is an Associate Professor at the Information Security Promotion Agency, Nagoya University. His research interests include computer vision and computer audition. He is a member of IEEE and RSJ.

Hiroaki Kudo received the degrees of B. Eng., M. Eng. and Dr. Eng. at Nagoya University, Japan in 1991, 1993 and 1996, respectively. In April 1996, he was a faculty member of the School of Engineering, Nagoya University as a Research Associate. In April 1997, he was a faculty member of the Graduate School of Engineering, Nagoya University. In April 1999, he was an Assistant Professor. In August 2000, he was an Associate Professor at the Center for Information Media Studies, Nagoya University. Since 2003, he has been a member of the Graduate School of Information Science, Nagoya University. He was a Research Fellow of the Japan Society for the Promotion of Science in 1995. His research interests include visual perception and computer vision. He is a member of IEEE, ITE, and IEEJ.



Noboru Ohnishi received the B. Eng., M. Eng. and D. Eng. degrees from Nagoya University, Nagoya, Japan, in 1973, 1975 and 1984, respectively. From 1975 to 1986 he was with the Rehabilitation Engineering Center under the Ministry of Labor of Japan. From 1986 to 1989 he was an Assistant Professor in the Department of Electrical Engineering, Nagoya University. From 1989 to 1994, he was an Associate Professor. Since 1994, he has been a professor in Nagoya University. From 1993 to 2001, he concurrently held the position of Head of Laboratory for Bio-mimetic Sensory Systems at the Bio-mimetic Control Research Center of RIKEN. He is now in the Graduate School of Information Science. His research interests include computer vision and computer audition, robotics, bio-cybernetics, and rehabilitation engineering. Dr. Ohnishi is a member of IEEE, IEEJ, IPSJ, SICE, JNNS, IIITE and RSJ.