Heliyon 6 (2020) e03670

Contents lists available at ScienceDirect

Heliyon

www.elsevier.com/locate/heliyon

Research article

A multi-objective approach for designing optimized operation sequence on binary image processing

Claudio Lezcano [a], José Luis Vázquez Noguera [a], Diego P. Pinto-Roa [a], Miguel García-Torres [b], Carlos Gaona [a], Pedro E. Gardel-Sotomayor [c,*]

[a] Facultad Politécnica, Universidad Nacional de Asunción, Paraguay
[b] Division of Computer Science, Universidad Pablo de Olavide, ES-41013 Seville, Spain
[c] CICTIA, Universidad Católica Nuestra Señora de la Asunción, Campus Alto Paraná, Paraguay

ARTICLE INFO

Keywords:
Computer science
Binary image processing
Operation sequence
Multi-objective optimization
Evolutionary algorithms

ABSTRACT

In binary image segmentation, the choice of the order of the operation sequence may yield to suboptimal results. In this work, we propose to tackle the associated optimization problem via multi-objective approach. Given the original image, in combination with a list of morphological, logical and stacking operations, the goal is to obtain the ideal output at the lowest computational cost. We compared the performance of two Multi-objective Evolutionary Algorithms (MOEAs): the Non-dominated Sorting Genetic Algorithm (NSGA-II) and the Strength Pareto Evolutionary Algorithm 2 (SPEA2). NSGA-II has better results in most cases, but the difference does not reach statistical significance. The results show that the similarity measure and the computational cost are objective functions in conflict, while the number of operations available and type of input images impact on the quality of Pareto set.

1. Introduction

The task of designing sequences of operation for binary image processing is not always trivial [1, 2]. Basically, it involves the selection of a set of logical and morphological operations with special shapes known as structuring elements [3]. To determine the number of structuring elements is not trivial, due to the number of possible combinations of structuring elements with different morphological operations. Designing an operation sequence with morphological operations is analogous to creating an instruction sequence for a computer program. A remarkable difference would be that each instruction comes from a large set of possible instructions unlike computer programming instructions [3]. The binary operation sequence problem consists in calculating a sequence of morphological operations that produces an output image I_{out} given an input image I_{in}. In the ideal case, the output image I_{out} is equal to I', where I' is a target binary image containing ideal features derived from I_{in}. Different sequences of morphological operations can obtain results equal to I'. The ideal would be to obtain the sequence of operations with the lowest computational cost. In many applications, one can find sequences of morphological operations with reasonable time that applied to I_{in} obtain images similar to I'. In this work, we look for different sequences of morphological operations that applied to I_{in} obtain images as similar as possible or equal to I' with low computational cost.

So far, the literature reports several works [4, 5, 6, 7] that apply heuristic techniques. However, none has proposed multi-objective heuristic approaches that are important indeed, since there are possibly multiple criteria in conflict. For example, several researchers have proposed the automation of morphological operation sequence design based on Genetic Algorithms (GA) [4, 6, 7, 8, 9] and Genetic Programming (GP) [5]. Quintana et al. have published several papers related to the generation of solutions using GP [3, 10, 11, 12]. In [3] pair images (original/target) are provided to guide the GP towards the desired feature extraction; in this context, a quality function measures the parameterizations in terms of the sensitivity and the specificity. For constructing morphological filters, 512 structuring elements are used in a regular and irregular way (randomly chosen), along with the basic operations of erosion and dilation.

Pedrino et al. have also published some works using GP for generating sequences of morphological operation [13, 14, 15]. In [15], a hardware architecture is presented for a solution based on the field of programmable gate arrays, to get the hardware architecture by means

* Corresponding author.
E-mail address: pedro.gardel@uc.edu.py (P.E. Gardel-Sotomayor).

https://doi.org/10.1016/j.heliyon.2020.e03670
Received 24 October 2018; Received in revised form 30 December 2019; Accepted 23 March 2020

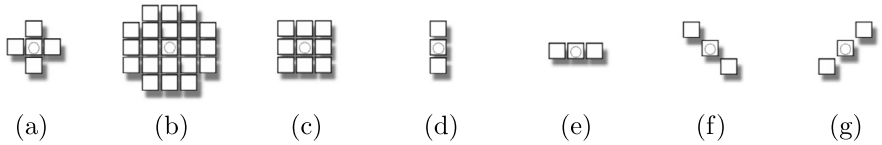


Fig. 1. Structuring Elements: a) cross, b) diamond, c) square, d) vertical line, e) horizontal line, f) main diagonal and g) secondary diagonal. Circles inside of blocks indicate the specific pixel over which operations are performed.

of reverse engineering, where the cost function is the average absolute error [16]. The program construction is implemented by 12 types of morphological operation sequences (pair morphological filter/structuring element), 6 types of logical operations, and reading and storing operations, respectively.

On the other hand, the literature reports several real life problems with several objective functions in conflict [16, 17]; i.e. the improvement of some objective function implies worsening others. In this context, to study the binary operation sequence design problem in a multi-objective context is crucial for real applications. Firstly, to understand the impact of different objective functions with each other; and secondly, to propose appropriate solution approaches considering all contradictory objective functions with equal importance.

Note that, the works [3, 10, 11, 12, 13, 14, 15] consider just one performance measure. On this basis, the solutions quality will be related to the number of available operations. So, we can consider, as a first approach to the multi-objective approach, the following objective functions: (a) the similarity measure between ideal and calculated output images by an operation sequence, and (b) the computational cost of calculating the output image.

Therefore, we can summarize the contributions of this paper as follows: (a) tackle the optimization problem by means of a multi-objective approach, (b) analyze the performance of two popular MOEAs strategies to calculate the set of non-dominated solutions, and (c) Study the advantages of the multi-objective approach by comparing the multi-objective results with a GA.

This paper is organized as follows. Section 2 introduces basic concepts of binary mathematical morphology and multi-objective optimization. Then, in Section 3, the proposed methodology is presented, followed by the experimental results in Section 4. Finally, the conclusions and future works are stated in Section 5.

2. Background

This subsection describes the binary mathematical morphology where the basic operators are defined and the multi-objective optimization problem where the solution vector of the problem addressed in this paper is presented.

2.1. Binary mathematical morphology

In the design process of operation sequence there are used several types of operations. In this work, logical and morphological operations are considered. Logical operations involve well known operations as: *or, and, exclusive or, binary complement, binary addition,* and *subtraction*; furthermore, *stacking operations*.

Typically, the geometrical structures of image are studied by the morphology. In this context, morphological operations aim to extract relevant structures from the image. In this work, a binary image I is defined as a mapping of a subset D_I from a domain definition Z^n of I in the pair $\{0,1\}$, that is:

$$I : D_I \subset Z^n \rightarrow \{0,1\}, \tag{1}$$

so that, the value of each pixel p of the image I is 0 or 1, i.e., $I(p) \in \{0,1\}\, \forall p \in D_I$.

Table 1
Basic operations.

Symbol	Description	Structuring element	Input image
dil_s	dilation	square	I_i
ero_s	erosion	square	
dil_c	dilation	cross	
ero_c	erosion	cross	
dil_h	dilation	horizontal line	
ero_h	erosion	horizontal line	
dil_v	dilation	vertical line	
ero_v	erosion	vertical line	
dil_sd	dilation	secondary diagonal	
ero_sd	erosion	secondary diagonal	
dil_md	dilation	main diagonal	
ero_md	erosion	main diagonal	
xor	exclusive OR		I_i and I_s
and	logical AND		
or	logical OR		
xnor	not exclusive OR		
sub	binary subtraction		
sub2	binary subtraction		I_s and I_i
store	temporary storage		I_i
ldi	original image reading		I_{in}
nop	no operation		I_i

Hadwiger defined the dual operation of Minkowski: addition and subtraction [18]. Matheron and Serra named the dual Minkowski operations with the current names of the two basic mathematical morphology operations: Dilation and Erosion [19, 20, 21].

A morphological operation requires a structuring element or kernel [12], in which a specific pixel is defined and the process performs on it. Fig. 1 shows the structuring elements used in this work.

The erosion of binary image I by structuring element B is denoted by $\epsilon_B(I)$ and is defined as the set of pixels p such that B is included in I when its origin is placed at p:

$$\epsilon_B(I) = \{p : B_p \subseteq I\}, \tag{2}$$

where B_p is the structuring element centered at $p \in I$.

Similarly, the dilation of binary image I that is done by a structuring element B is denoted by $\delta_B(I)$. Such operation is defined as a set of pixels p such that B reaches I when its origin coincides with p:

$$\delta_B(I) = \{p : B_p \cap I \neq \emptyset\}. \tag{3}$$

The difficulty of understanding *a priori* the impact over the performance of solution, by using different structuring elements, operation sequences and the application order of these operations, makes morphological operations design problems an ideal area to apply evolutionary algorithms [22, 23, 24].

2.2. Morphological operation set

In this paper the morphological operation set t is the set containing all the valid operations that can be used. In Table 1 the basic morphological operations considered in this paper are listed.

Extra operations. In addition to the basic operations listed in Table 1, erosion and dilation with another structuring elements and stacking operation are included for more advance tests. Table 2 shows these operations. The computational costs of applying the basic and extra morphological operations are presented in Table 3.

Table 2
Extra operations.

Symbol	Description	Structuring element	Input image
ero_ro	erosion	rhombus	I_i
dil_ro	dilation		
ero_d	erosion	diamond	
dil_d	dilation		
ero_sq	erosion	square (5×5)	
dil_sq	dilation		
push	stacking	-	
pull			

2.3. Multi-objective optimization problem

The Multi-objective Optimization Problem (MOP) can be expressed as follows [17]:

$$\min z = f(x) = \left[f_1(x), f_2(x), ..., f_F(x)\right], \tag{4}$$

subject to:

$$h(x) = \left[h_1(x), h_2(x), ..., h_k(x)\right] = 0, \tag{5}$$

$$g(x) = \left[g_1(x), g_2(x), ..., g_l(x)\right] \geq 0, \tag{6}$$

where $x \in S$, x is the decision vector, S is the set of solutions and $f_i(x)$ is the i-th objective function of the problem. The problem constraints are represented by $h(x)$ and $g(x)$.

In our problem, a solution is represented as a numeric vector x as depicted in Fig. 2, where each x_i belongs to an operation of the set t as defined in Tables 1 and 2.

The set of feasible solutions S_{fac} consists of decision vectors that satisfy the equality $h(x)$ and inequality $g(x)$. Therefore:

$$S_{fac} = \{x \in S \ : \ g(x) \geq 0 \ \wedge \ h(x) = 0\}. \tag{7}$$

Any decision vector $x \in S_{fac}$ defines a feasible solution for the problem in question.

The subset of S_{fac} that minimizes the decision vector $f(x)$ is the Pareto set S^*, where its projection on the objective space is the Pareto front $FP = \{f(x) : x \in S^* \subseteq S_{fac}\}$. The goal of a pure multi-objective optimization process is to calculate the optimal Pareto set S^*.

Multi-objective Optimization in the Pareto context implies that: given two solutions $x_a, x_b \in S_{fac}$ the following conditions have to be satisfied:

$$x_a \succ x_b \ \texttt{iff} \ \forall i : \ f_i(x_a) \leq f_i(x_b) \ \wedge \ \exists j : \ f_j(x_a) < f_j(x_b), \tag{8}$$

where $x_a \succ x_b$ indicates that x_a dominates x_b.

If S is a finite set, then the problem is a combinatorial problem. MOP can be approached in two ways: classical or mono-objective optimization and pure or Pareto multi-objective optimization. In the first case, the algorithm obtains just one solution. There are several techniques reported in [17]. These techniques need information about the structure of the problem, in advance, to define priority among objective functions; but, the structure is usually unknown.

In pure multi-objective optimization, algorithms calculate a set of non-dominate solutions (Pareto set) where all objective functions are considered with the same importance level. Moreover, the Optimal Pareto front describes the structure of the frontier in the criteria space, where all solutions are optimal in the Pareto context. By analyzing this structure, the decision maker can select the most appropriated solution according to the business needs. Of course, any optimal solution obtained by a mono-objective algorithm is part of the Pareto set.

Several image processing problems can be solved in a multi-objective context, for example the multi-objective image coding [25], the multi-objective change detection in multispectral images [26], multi-objective image segmentation [27, 28].

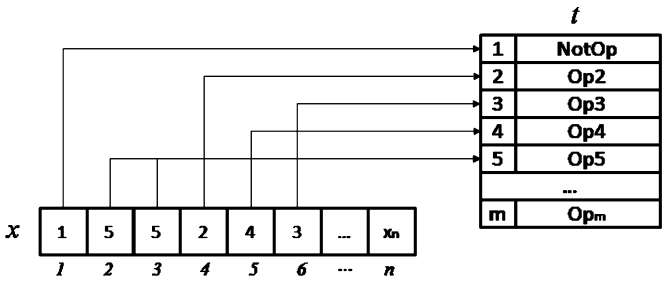


Fig. 2. Relationship between chromosome and table of operations.

3. Materials and methods

In this section we describe the Multi-objective Operation Sequence Design (MOSD) Problem, and the Genetic Algorithm and Evolutionary proposals.

3.1. Multi-objective formulation

The Multi-objective Operation Sequence Design (MOSD) problem consists of calculating a solution x that allows to obtain an image $I_{out} \approx I'$ by applying x to I_{in}, so that, the similarity measure f_1 between I_{out} and I' and the computational cost f_2 are simultaneously minimized. That is,

$$\min z = f(x) = \left[f_1(x), f_2(x)\right], \tag{9}$$

subject to:

$$x_i \in \{1, 2, \ldots, m\} \ \forall i, \tag{10}$$

where:

- $f_1(x) = d(I_{out}, I')$ is the Image similarity measure that will be explained below,
- $f_2(x) = C(x)$ is the computational cost to calculate I_{out} by a solution coded in x,
- $x = \left[x_1, x_2, \ldots, x_n\right]$ is the vector solution of n elements that represents a sequence of instructions t_{x_i} of n steps.

In this context, each x_i element indexes an operation of the set t. Without loss of generality, I_{out} is obtained by the operation sequence as follows: $I_1 = t_{x_1}(I_{in})$, $I_2 = t_{x_2}(I_1)$, …, $I_{out} = t_{x_n}(I_{n-1})$, where $I_i = t_{x_i}(I_{i-1})$ is the i-th intermediate image calculated by applying the operation t_{x_i} on the previous image I_{i-1}. Note that $I_0 = I_{in}$ and $I_n = I_{out}$, and t_{x_i} is the x_i-th operation of list t that works on input image I_{i-1} to calculate image I_i at the i-th step.

Image similarity. The similarity measure $d(I_{out}, I')$ is quantified using the following formula [29]:

$$d(I_{out}, I') = 1 - \frac{TP}{TP + FN + FP} \quad , \tag{11}$$

where TP is the number of true positives, FN is the number of false negatives, and FP is the number of false positives.

Let D be the domain of images I_{out} and I', and p any point in D, then TP, FN and FP can be calculated as:

$$TP = \sum_{p \in D} (I'(p) \cdot I_{out}(p)), \tag{12}$$

$$FN = \sum_{p \in D} (I'(p) - [I'(p) \cdot I_{out}(p)]), \tag{13}$$

$$FP = \sum_{p \in D} (I_{out}(p) - [I_{out}(p) \cdot I'(p)]), \tag{14}$$

The similarity measure (11) was selected to yield good results as a measure of similarity between sets according to [30]. In this case, the lower the similarity measure value is, the more similar the images are.

Table 3
The costs of applying operations.

Operation	Structuring element	Cost
Erosion, dilation	cross	5
	rhombus	13
	diamond	21
	square 5x5	25
	square 3x3	9
	vertical	3
	horizontal	3
	main diagonal	3
	secondary diagonal	3
Logical operations	–	1
Storage and Read operations	–	1
No operation	–	0

Procedure *Genetic Algorithm*
begin
1: *GeneratePopulation* (P_k);
2: *valuatePopulation* (P_k);
3: **repeat**
4: *SelectParents* (P_k, P'_k);
5: *ApplyOperators* (P'_k, P''_k);
6: *EvaluatePopulation* (P''_k);
7: *UpdatePopulation* (P''_k, P_k);
8: **until** (*StoppingCriterion*)
end

Fig. 3. Genetic Algorithm pseudocode. P_k is the current population at generation k, P'_k is the set of solutions for combination and P''_k the new solutions generated.

Computational Cost. The goal of this criterion $C(x)$ is to measure the number of operations to calculate I_{out} by the solution x considering as initial image I_{in}. This cost is obtained by the following equation:

$$C(x) = \sum_{i=1}^{n} cost(t_{x_i}), \tag{15}$$

where $cost(t_{x_i})$ is the cost of applying the operation t_{x_i}, according to Table 3.

3.2. Algorithms

In this subsection we present the algorithms implemented and compared in this work.

3.2.1. Genetic algorithm

The Genetic Algorithm (GA) was first presented by Holland [31]. It is an evolutionary population-based strategy that uses evolutionary biology-based techniques such as inheritance, mutation, selection, and crossover. This strategy is widely used in many optimization problems [32, 33], including the operation sequence problem.

In GA, each solution is encoded by chromosome and represents an individual in the GA context. Chromosome is composed of genes, which are digits in the solution. The alleles are the possible values a gene can take. Binary encoding, which is one of the most used methods, consists of individuals represented as binary arrays.

Fig. 3 shows the pseudocode of the GA. It starts by generating randomly a population of solutions. The most promising individuals, according to a selection procedure, are selected to generate new ones after applying genetic operators such as crossover and mutation. The purpose of crossover is to combine parents to generate new offsprings. Mutation produces some changes on a single individual and introduces diversity in the population. The process continues across several generations until a stopping criterion is reached.

The efficiency of a GA is greatly dependent on its tuning parameters. We set the crossover probability to 0.9 and the mutation probability to 0.1 following the recommendations of [13]. In order to set the population size and the number of iterations, we have to take into account that in problems with a very large solution space the population size must be large enough to obtain a representative sample of the solution space. Furthermore, a very small population may result in premature convergence, whereas a very large population may result in a slow convergence rate. Since both values are dependent on the problem, we conducted several experiments with different values for the population size and the number of iterations. After analyzing convergence and computational time, we fixed the population at 50 individuals and the number of iterations was set to 40,000.

Table 4
Evolutionary parameters.

Parameters	Values
Population size	50
Type of crossover operation	One-point crossover
Crossover probability	0.9
Type of mutation operation	Flip Bit Mutation
Mutation probability	0.1
Type of selection operation	Binary tournament
Number of operations m	21 (sections 4.2 and 4.1), 28 (section 4.3)
Number of generations	40,000 (sections 4.2 and 4.1), 400,000 (section 4.3)

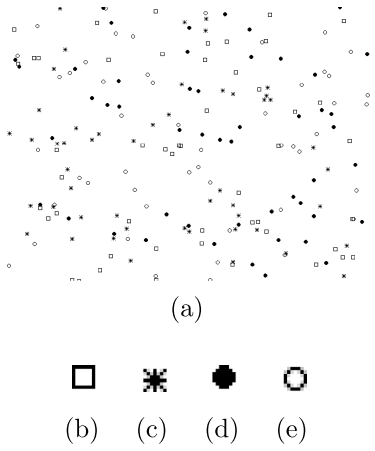


Fig. 4. (a) Input image I_{in} is conformed by (b) squares, (c) stars, (d) circles, and (e) rings.

3.2.2. NSGA-II

NSGA-II [34] is a population based Multi-Objective Evolutionary Algorithm [17] (MOEA) that has become very popular in real and theoretical problems and, therefore, has been established as a benchmark in the literature.

This strategy starts generating and initial random population P_0 of N solutions. Then, solutions are sorted according to non-dominance and a rank value is associated with them so that first, with rank 1, we can find non-dominated solutions. Next, with rank 2 we find those solutions that belong to the next non-dominated front and so on. At each iteration k the strategy generates an offspring population Q_k of N solutions mating solutions by applying the binary tournament selection and using the crossover and mutation operators. Then, both populations, parents P_k and offspring population Q_k, are combined to generate the new population $P_{next} = P_k \cup Q_k$. The next parent population P_{k+1} is generated by deleting, from P_{next}, the worst N solutions, i.e. parents and offspring compete to survive. The strategy stops after a fixed number of iterations.

The Binary Tournament Selection Method is used for the selection process. Four individuals are randomly chosen from the population and the top two are finally selected for crossing [35].

Table 5
NSGA-II and SPEA2 comparison considering hypevolumne difference with Known Optimal Pareto Front.

n		NSGA-II		SPEA2		p-value
		μ	σ^2	μ	σ^2	
30	Rings	**0.846148922**	0.002202403	0.859407178	0.001923174	0.262963097
	Circles	**0.721685031**	0.004815378	0.737396755	0.003214772	0.341089765
	Squares	**0.825396743**	0.005633618	0.848927581	0.004510275	0.20584782
	Stars	**0.875703847**	0.003536328	0.901736699	0.002775871	0.078004393
40	Rings	**0.918079604**	0.001306054	0.933393839	0.002167348	0.160418485
	Circles	0.777907299	0.004249807	**0.768206417**	0.003807213	0.556225212
	Squares	**0.887484025**	0.002473421	0.911671216	0.00302404	0.079295151
	Stars	**0.9510287**	0.00085453	0.96048867	0.001856782	0.324392428
50	Rings	**0.993579178**	0.000121602	0.994237983	0.000160885	0.830786378
	Circles	**0.939265028**	0.002620776	0.948524838	0.001655266	0.441303367
	Squares	0.984098363	0.000746718	**0.837577391**	0.006172815	2.14897E-11
	Stars	**0.998035439**	1.68425E-05	0.998612133	3.75962E-05	0.670412064

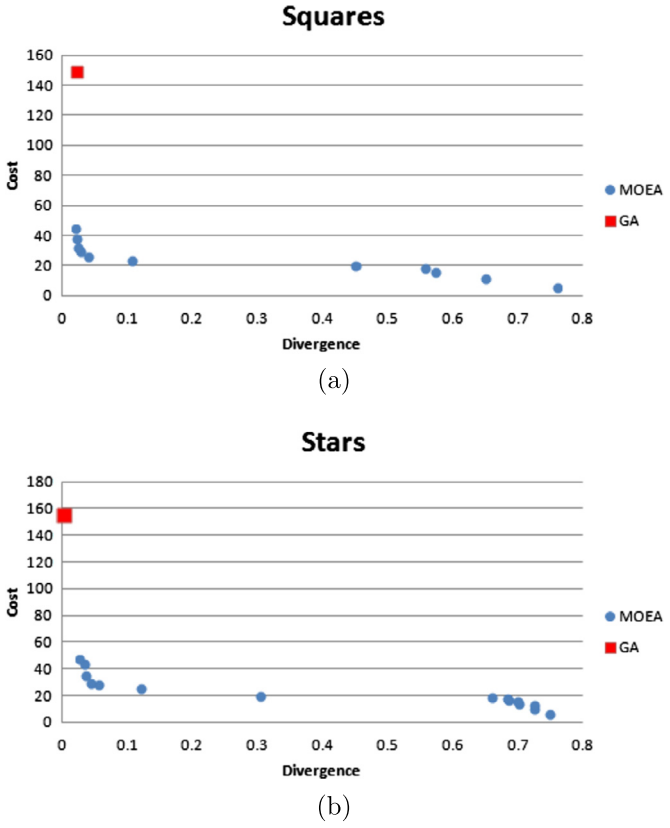


Fig. 5. Pareto front and solution calculated by MOEA(NSGA-II) and GA using operations of Table 1, considering Squares and Stars.

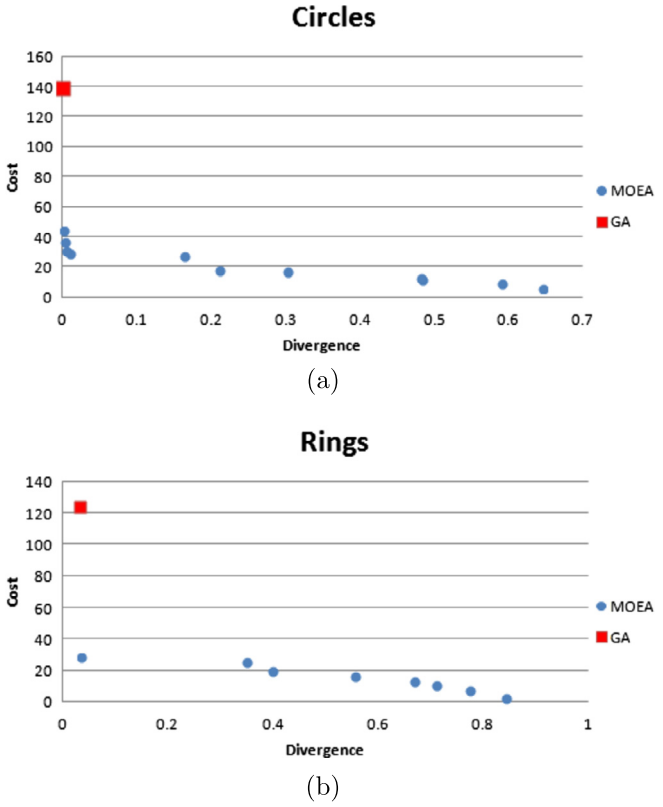


Fig. 6. Pareto front and solution calculated by MOEA(NSGA-II) and GA using operations of Table 1, considering Circles and Rings.

The crossover operation is based on the One-point crossover [35]. Through this method a single-point crossover is selected randomly in the chromosome to carry out the crossing. The new individual is made up by the two elements of the first one selected up to the One-point crossover, then it is completed with the elements of the second individual chromosome. The resulting chromosome is a new solution.

The mutation operation is a variation of the Flip-Bit Mutation Method, where an gen of the individual submitted for mutation is randomly chosen and such gen is altered by another value randomly chosen between 0 and m [35].

3.2.3. SPEA2

SPEA2 is an improved version of SPEA [36], and was proposed by Zitler et al. [37]. This strategy is also widely used in multi-objective optimization problems.

This strategy follows an evolutionary scheme similar to NSGA-II. This algorithm follows the original idea in which solutions are ranked based on a fitness function for each individual. Such fitness value is computed according to the number of solutions that are dominated and the number of solutions it dominates in a population. In addition, equivalent solutions are ranked based on diversity. In contrast to SPEA, SPEA2 incorporates a more fine-grained fitness assignment strategy, a density estimation technique, and an enhanced archive truncation method.

4. Results

For this study the *image segmentation problem* is selected as a particular case of MOSD problem. This experimentation is carried out in four different tests.

In the first experiment, we want to determine:

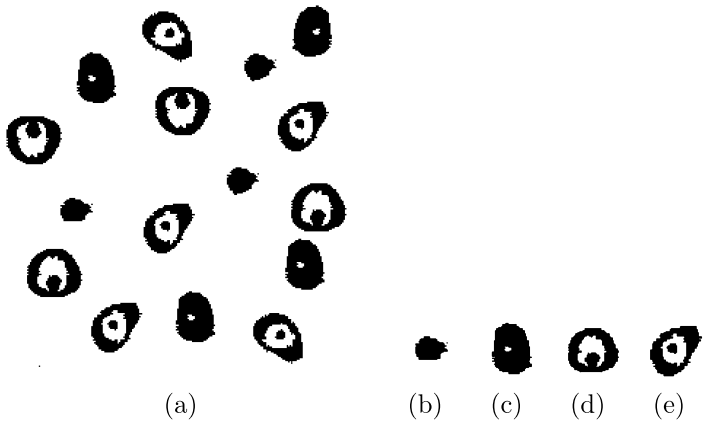


Fig. 7. (a) Binary Image of cells I_{in}, (b) type 1 cell, (c) type 2 cell, (d) type 3 cell, and (e) type 4 cell.

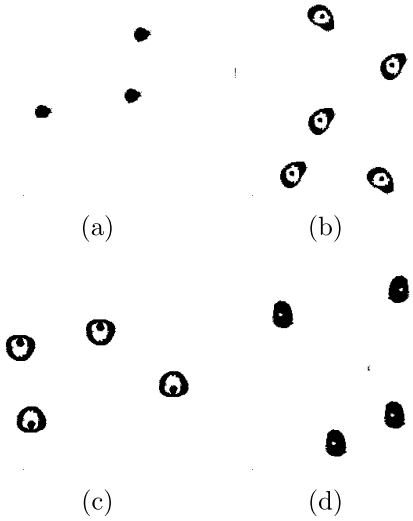


Fig. 8. Target Images (a) type 1 cell, (b) type 2 cell, (c) type 3 cell, and (d) type 4 cell.

- which MOEA is more suitable for this problem, and
- the appropriated chromosome length value, n. For that, we consider two of the most important MOEA proposed in the literature [17]: SPEA2 and NSGA-II and n values of 30, 40 and 50. Both MOEAs and n value performance are compared by Hipervolumen metric [17].

In the second experiment, given that the GA-based algorithm is the main solution reported in the state-of-the-art for image segmentation problem [13], we study strengths and weaknesses of GA solutions versus the MOEA Pareto front.

The third experiment studies the Pareto Front quality versus operation list.

Finally, in the fourth test we study the MOSD problem considering two different real images with the method this paper proposes, the results show the advantages of the methodology and possible applications to real life problems.

All algorithms were coded in Java, programming language using the jMetal framework [38], and performed on a Dual Intel Xeon CPU L5420 @ 2.50 GHz processor with 4 cores each, having 24 GB of RAM available, with all its resources available for the execution of test cases. All tests were performed considering input image I_{in} and target image I' and the evolutionary/genetic parameters are similar to those suggested by [23, 39, 40], see Table 4.

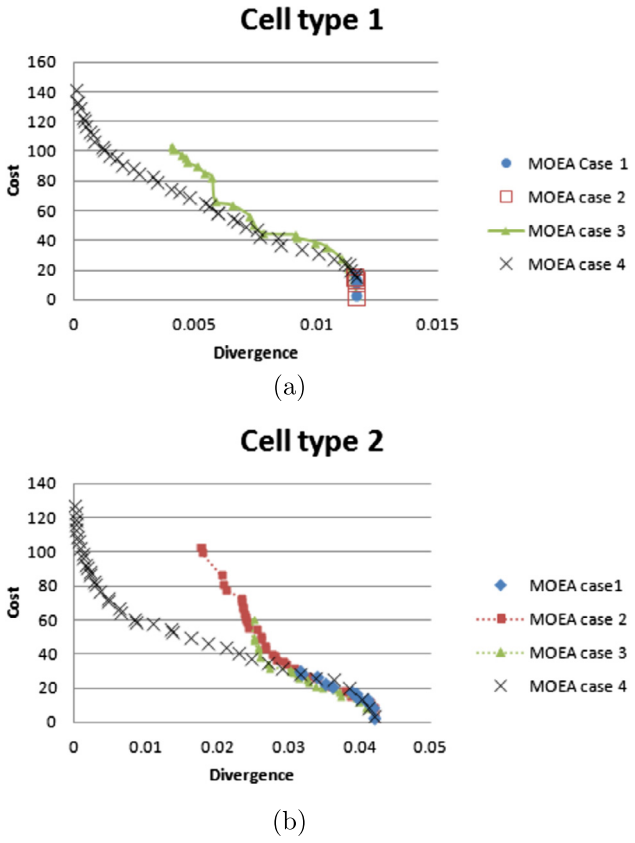


Fig. 9. Pareto front and solution calculated by MOEA (NSGA-II) using operation given in Tables 1 and 2, considering Cell type 1 and 2.

4.1. Determination of the Chromosome length and comparison between SPEA2 and NSGA-II performance

Test Image. In this test, four training images were considered. A synthetic image has been generated for this purpose as shown in Fig. 4, for this figure the input image I_{in} is composed of four distinct elements: squares, circles, stars, and rings. All of these images are randomly distributed. The target image I' and input image I_{in} are similar to each other, though I' only contains one of the four elements.

Operations. The set t contains the next operations: morphological, logical, reading, and storing. This set is listed in Table 1, and is based on [5].

Independently of the image size and morphological operations, the computational cost is defined simply by the number of active com-

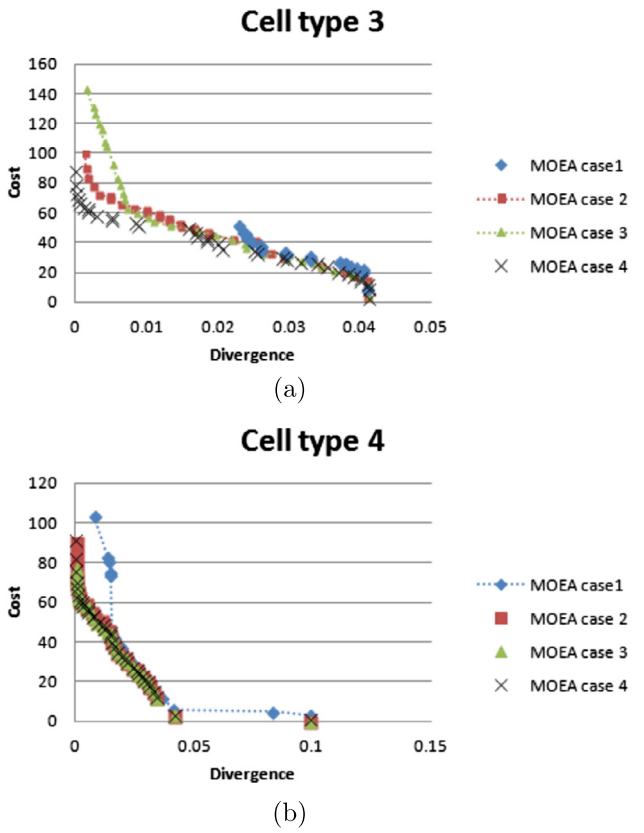


Fig. 10. Pareto front and solution calculated by MOEA (NSGA-II) using operation given in Tables 1 and 2, considering Cell type 3 and 4.

ponents that conforms a structuring element (SE) for morphological operations. For logical operations, the cost is just one unit and normalized. Table 3 shows the cost of structuring elements given in Fig. 1 and logical operations.

Results. The hypervolume measures the area of a Pareto front considering a reference point [41]. This reference point usually is the worst point of criteria space. When the hypervolume is higher the Pareto front is better quality. The higher hypervolume possible corresponds to the Optimal Pareto Front.

Considering the importance of the Chromosome length, n, in the algorithm performance a tuning procedure to find the best value for this parameter was carried out. The procedure was implemented in the following way. For every image in Fig. 4 each MOEA was run 30 times to get 30 Pareto Fronts, using all other evolutionary parameters as shown in Table 4. The set of non-dominated solutions found in all executions of both MOEAs is denominated as the Known Optimal Pareto Set. For each Pareto Front, generated by a single algorithm execution, was calculated the difference between its hypervolumen and the hypervolumen of the Known Optimal Pareto Front, therefore, the lower the difference, the closer the Pareto front is to the optimal one. Next, for each image and MOEA was calculated the average and variance considering the 30 hypervolumens differences. The same procedure was performed considering three different Chromosome length n values, 30, 40 and 50. The results presented in Table 5 show that the MOEAs presented a better performance with a n value of 30. The best result of each MOEA, when different n values are considered is underlined, for example, the best performance of the NSGA-II considering the Rings image was obtained with n equal to 30 with an average of 0.846148922. In all four cases, considering Rings, Circles, Squares and Stars, the best performance of the MOEAs was achieved with a n value of 30 with a high statistic significance (p-value < 0.05 in all cases), therefore, all following tests were done considering a chromosome length of 30. Once the evolutionary parameters for the tests are defined the question about which MOEA shows a better performance for this problem can be evaluated. The results presented in Table 5 also show the performance comparison between MOEAs. Note in the Table 5 rows indicating n values and figures while columns contain average (μ) and variance (σ^2) of MOEA algorithms hypervolumen difference with the optimal, and the p-value of the t-test statistic related to the comparison of MOEAs with same figure and Chromosome length. There the best result of each line is highlighted in bold. The p-value shows that there is not a statistically significant difference between MOEAs performance, therefore, any of them could be considered for the MOSD problem. In consequence, we will just consider the NSGA-II approach for next experiments.

4.2. Analysis of GA and NSGA-II performance

A GA algorithm developed in [4] was implemented to compare the performance of the NSGA-II. This GA algorithm just optimizes the similarity measure.

For each pair of images, the NSGA-II algorithm was performed 10 times to obtain 10 Pareto sets S_k. Then, a good Pareto set S^* was built firstly by the union of these sets in $S' = \cup_{k=1}^{10} S_k$, and then dominated solutions are eliminated, i.e. $S^* = S' - \{x \in S' : S' \succ x\}$. The same procedure is done for GA were the best from 10 calculated solutions is taken.

Test Image. In this test, the images given in the Fig. 4 are considered. Similar to the previous experiment, each input image I_{in} is composed of four different elements: squares, circles, stars, and rings. All of these images are randomly distributed. The target image I' and input image I_{in} are similar to each other, though I' only contains one of the four elements.

Operations. The set t contains the next operations: morphological, logical, reading, and storing and listed in Table 1.

Results. In this experiment the algorithms generated solutions with zero similarity measure values as well as low run-time operation sequences as shown in Figs. 5 and 6; the Pareto front $f(S^*)$ of NSGA-II and a single solution of GA are exposed respectively. Note that, NSGA-II calculates several solutions while GA just one solution, i.e. NSGA-II is a pure multi-objective optimization approach and GA is a mono-objective solution.

In the first test, target image I' contains only square elements. We find the GA solution is inside the Pareto front with a very low similarity measure value, but a high computational cost, as seen in Fig. 5a. Similar results are obtained in all tests, as see in Figs. 5b, 6a and 6b with target image I' containing only stars and circle elements respectively. In all cases, NSGA-II calculates solutions with lower, intermediate, and high values of computational cost and similarity measure.

4.3. Impact of operations on Pareto front quality

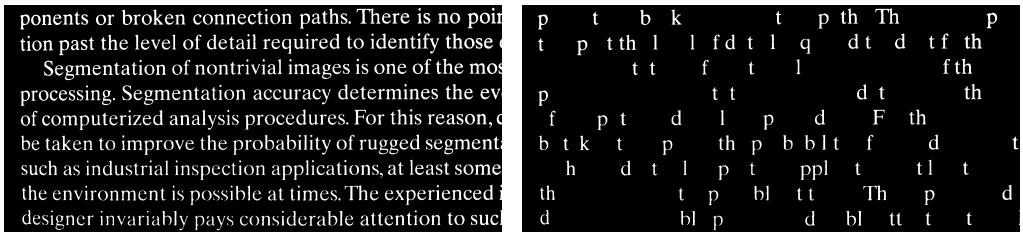
Test Images. Similar to the first experiment (section 4.1), for this test a synthetic image has been generated as shown in Fig. 7. In this case the pair of binary images I_{in} were artificially created by means of four distinct types of cells, all of these randomly distributed, as shown in Fig. 7. The target image I' contains only one of the element types which are in the same position than in the original image I_{in}.

Operations. In addition to the operations used in the previous test, erosion and dilation with another structuring element and stacking operation are included. Table 2 shows new operations.

Results. The target of this experiment is to isolate each of four types of cells as given in the Fig. 8.

The experiment of this group was performed in four consecutive steps.

- In this first step, the same evolutionary parameters and the set of operations t of previous experiment are used (Table 1). Fig. 9 shows the different Pareto fronts for the four cells of the Fig. 7. Note that,



(a) (b)

Fig. 11. (a) Input image of the Text example and (b) target image I' containing only the predetermined characters.

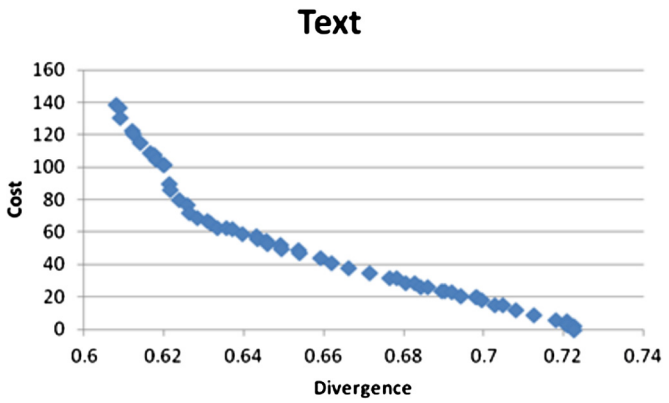


Fig. 12. Pareto front calculated by the NSGA-II considering the Text instance.

all solutions of MOEA case 1 have high similarity measure values. In order to improve these results, the next step is implemented.

- In the second step, the number of evaluations was increased from 40,000 to 400,000 generations through which a set of better solutions was obtained, but such solutions are still not satisfactory. In Fig. 9 these solutions are given by MOEA case 2. Similar to the results of the previous step, the solutions still have a high similarity measure.
- In the third step, all operations of Table 2 without *push* and *pull* were added to the set of operations t. As a result, solutions slightly improve specially for cell type 3 and 4, but for cell type 1 and 2 the similarity measure is still high. In Fig. 9 these solutions are given by MOEA case 3.
- In the last step, the *push* and *pull* operations were added, see the last operation of Table 2. With this complete set of operations (Tables 1 and 2), satisfactory solutions are obtained where one solution reaches zero similarity measure or is close to zero for all types of cells. In Fig. 9 and 10, these solutions are given by MOEA case 4. Note that, Pareto fronts of MOEA case 2 to 4 are very similar, as seen in Fig. 10b.

In all cases, the single-solution of GA is dominated by solutions of the Pareto front of MOEA case 4. Note that, all Pareto fronts of MOEA from cases 1, 2 and 3 are dominated by the Pareto front of MOEA case 4.

4.4. Real images problems

In order to validate the methodology proposed in real life applications, two instances of real images are considered. The first instance is proposed in this document as a simple application. The input image is a text [42] and the output image contains only predetermined characters of the original image (see Fig. 11). Fig. 12 presents the diversity of solutions achieved by the proposed methodology calculated, ranking from low to high cost.

The second instance uses a binary image of a music score as proposed by Yoda [9]. The goal of this second instance is to obtain a target image I' from the music score with only the Heads (Fig. 13(b)), Hooks (Fig. 13(c)) and Staff lines (Fig. 13(d)) of the original image (Fig. 13(a)). The Pareto fronts generated by the NSGA-II can be seen in Fig. 14. There can be seen that the problem considering the heads as target image I' generates images with lower similarity measure than the other two, but with higher costs.

Three examples of the images generated by the proposed methodology can be seen in Fig. 15.

5. Conclusions and future work

The Multi-objective Operation Sequence Design (MOSD) problem can be considered in several digital image problems. In this article, we present a multi-objective solution mechanism that searches for a sequence of steps that, given an input binary image, obtains an output binary image at the lowest computational cost. It is clear that the problem is a multimodal optimization problem, where we can get several trade-off solutions. We compare the performance of two Multi-objective Evolutionary Algorithms (MOEA): the Non-dominated Sorting Genetic Algorithm (NSGA-II) and the Strength Pareto Evolutionary Algorithm 2 (SPEA2). The NSGA-II shows better results in most cases but the difference is not statistically significant. The results show that: (a) the similarity measure and the computational cost are objective functions in conflict, therefore, the Pareto set contains several trade-off solutions; while (b) the number of operations available and type of input images impact on the quality of Pareto set. A Genetic Algorithm (GA) was compared to the performance of the NSGA-II. This GA algorithm just optimizes the similarity measure. There are several aspects to highlight. First, the number of different operations is more critical than the number of evolutionary generations for obtaining a good performance. This is because the number of possible solutions increases with the number of operations and consequently better solutions can be reached. Second, the MOEA solution with lowest similarity measure has lower cost than the GA solution in all cases. This indicates that the MOSD problem has several solutions with the same similarity measure, but different computational cost. In other words, the MOSD is a multi-modal optimization problem. In order to validate these results, two real problems were solved. The results show the capacity of the methodology to be used not only in images designed in the laboratory but also in real cases. The selection of a solution depends on the application scenario, for example, if the application needs a fast answer, then the selected solution will give non good similarity measure. In counterpart, if the application needs good similarity measure then the selected solution will have a higher computational cost. This proposal is the first part of this research line and we plan to extend it to more complex images and to gray-scale and color images. Approaches by other meta-heuristic algorithms should be explored, as well as adding other operations such as rotation, translation, change of scale, and others.

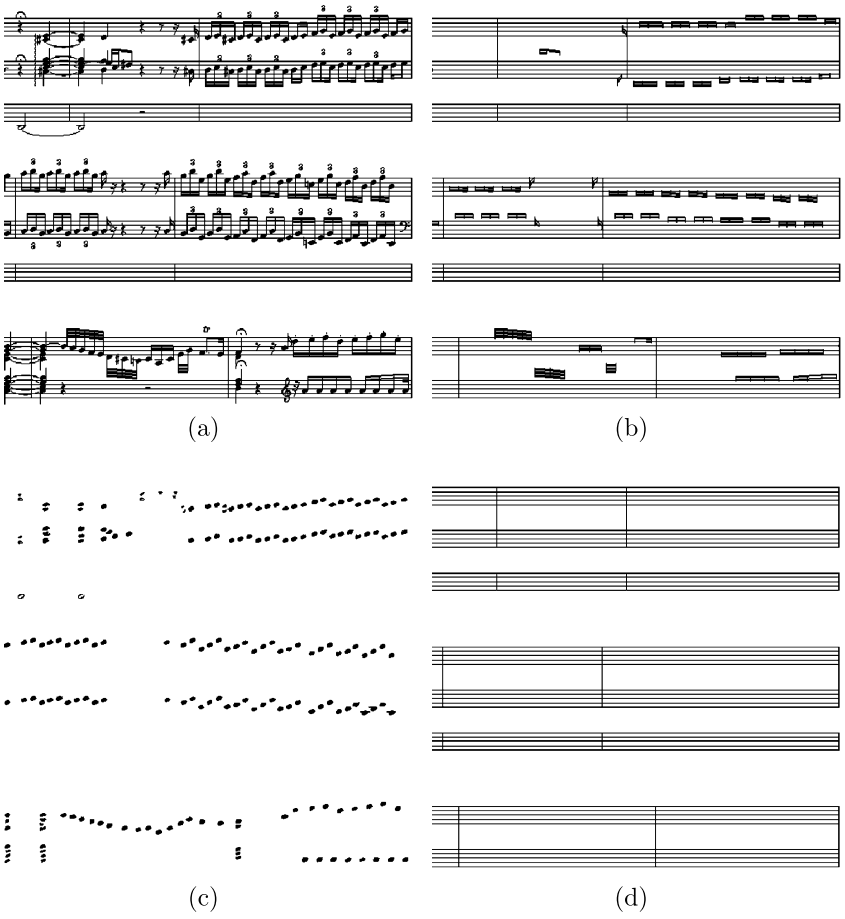


Fig. 13. (a) Original music sheet image, target image I' containing (b) the Hooks, (c) the Heads and (d) the Staff lines.

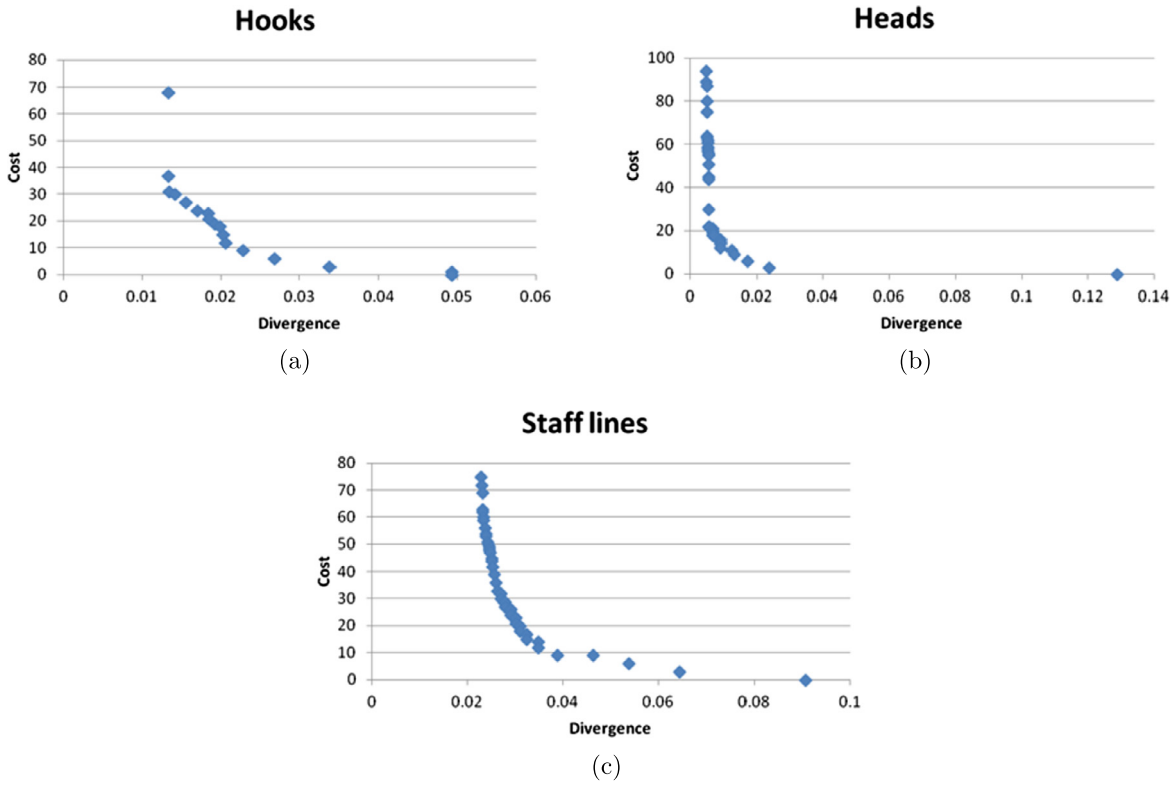


Fig. 14. Pareto fronts of the second instance, considering the target images I' as (a) the Hooks, (b) the Heads and (c) the Staff lines.

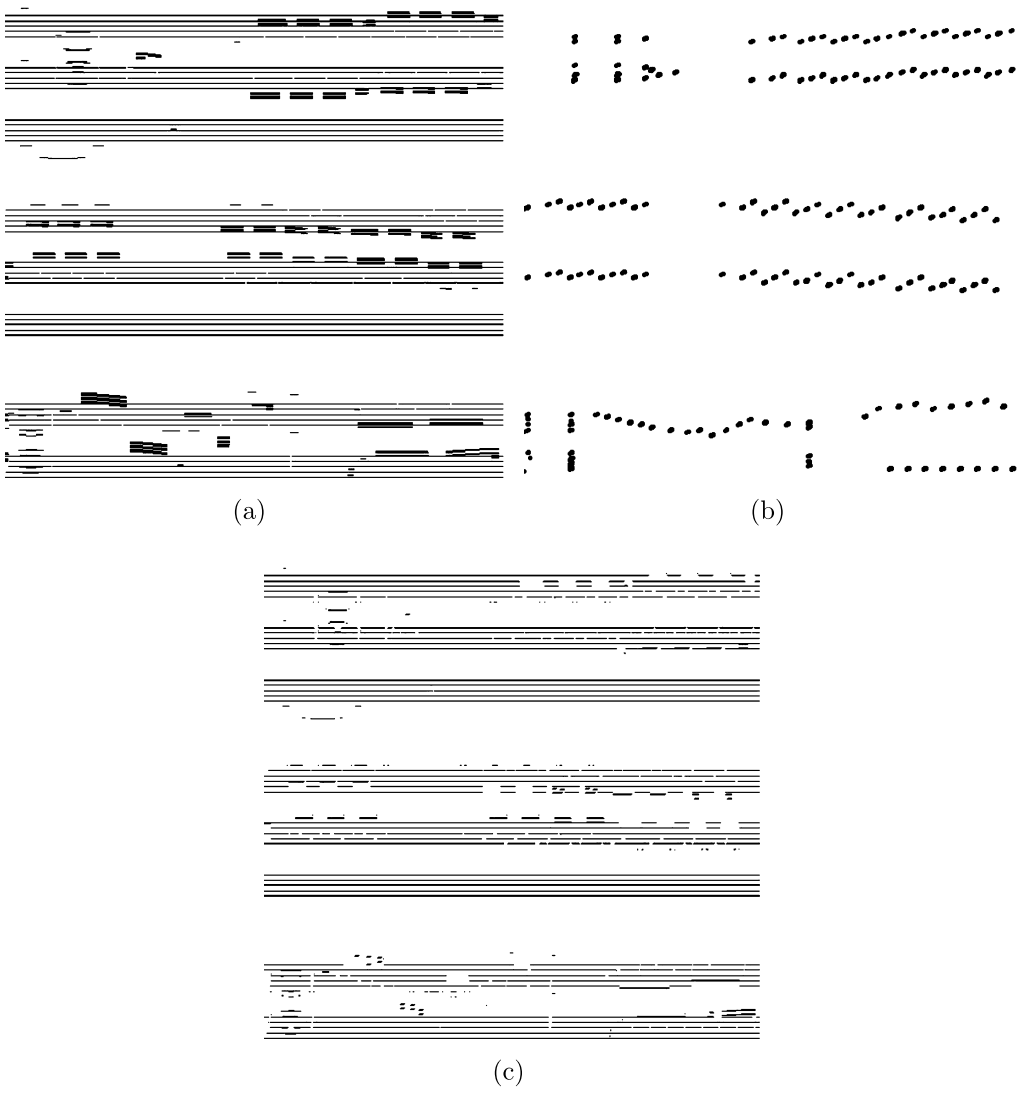


Fig. 15. Examples of the images generated by the methodology, considering the target images I' as (a) the Hooks, (b) the Heads and (c) the Staff lines 15c.

Declarations

Author contribution statement

Claudio Lezcano: Performed the experiments.

José Luis Vázquez Noguera, Diego P. Pinto-Roa: Conceived and designed the experiments; Analyzed and interpreted the data.

Miguel García-Torres: Analyzed and interpreted the data.

Carlos Gaona: Performed the experiments; Contributed reagents, materials, analysis tools or data.

Pedro E. Gardel-Sotomayor: Analyzed and interpreted the data; Wrote the paper.

Funding statement

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Competing interest statement

The authors declare no conflict of interest.

Additional information

No additional information is available for this paper.

References

[1] J. Barrera, G.J.F. Banon, E.R. Dougherty, Automatic Design of Morphological Operators, Springer New York, New York, NY, 2005, pp. 257–278.

[2] N.S.T. Hirata, E.R. Dougherty, J. Barrera, Iterative design of morphological binary image operators, Opt. Eng. 39 (2000) 3106–3123.

[3] M.I. Quintana, R. Poli, E. Claridge, Morphological algorithm design for binary images using genetic programming, Genet. Program. Evol. Mach. 7 (2006) 81–102.

[4] T.R. Crimmins, W.M. Brown, Image algebra and automatic shape recognition, IEEE Trans. Aerosp. Electron. Syst. AES-21 (1985) 60–69.

[5] J.H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, MIT Press, Cambridge, MA, USA, 1992.

[6] M. Schmitt, Mathematical morphology and artificial intelligence: an automatic programming system, Signal Process. 16 (1989) 389–401.

[7] R.C. Vogt, Automatic Generation of Morphological Set Recognition Algorithms, Springer-Verlag New York Inc., 1989.

[8] M. Liu, Y. Liu, H. Hu, L. Nie, Genetic algorithm and mathematical morphology based binarization method for strip steel defect image with non-uniform illumination, J. Vis. Commun. Image Represent. 37 (2016) 70–77, http://www.sciencedirect.com/science/article/pii/S104732031500067X.

[9] I. Yoda, K. Yamamoto, H. Yamada, Automatic acquisition of hierarchical mathematical morphology procedures by genetic algorithms, Image Vis. Comput. 17 (1999) 749–760, http://www.sciencedirect.com/science/article/pii/S0262885698001516.

[10] M.Q. Hernandez, Genetic programming applied to morphological image processing, Ph.D. thesis, School of Computer Science, University of Birmingham, 2005.
[11] R.P. Marcos, I. Quintana, E. Claridge, On two approaches to image processing algorithm design for binary images using gp, in: Applications of Evolutionary Computing, Springer, 2003, pp. 422–431.
[12] M.I. Quintana, R. Poli, E. Claridge, Morphological algorithm design for binary images using genetic programming, Genet. Program. Evol. Mach. 7 (2006) 81–102.
[13] E.C. Pedrino, O. Ogashawara, V.O. Roda, Reconfigurable architecture for mathematical morphology using genetic programming and fpgas, in: 2010 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum, IPDPSW, IEEE, 2010, pp. 1–4.
[14] E.C. Pedrino, J.H. Saito, V.O. Roda, Architecture for binary mathematical morphology reconfigurable by genetic programming, in: Programmable Logic Conference, SPL, 2010 VI Southern, IEEE, 2010, pp. 93–98.
[15] E.C. Pedrino, J.H. Saito, V.O. Roda, A genetic programming approach to reconfigure a morphological image processing architecture, Int. J. Reconfigurable Comput. 2011 (2011) 5.
[16] L.G. Moré, M.A. Brizuela, H.L. Ayala, D.P. Pinto-Roa, J.L.V. Noguera, Parameter tuning of clahe based on multi-objective optimization to achieve different contrast levels in medical images, in: 2015 IEEE International Conference on Image Processing, ICIP, 2015, pp. 4644–4648.
[17] C.C. Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, Springer, 2007.
[18] H. Hadwiger, Minkowskische addition und subtraktion beliebiger punktmengen und die theoreme von erhard schmidt, Math. Z. 53 (1950) 210–218.
[19] G. Matheron, J. Serra, The birth of mathematical morphology, in: Proc. 6th Intl. Symp. Mathematical Morphology, Sydney, Australia, 2002, pp. 1–16.
[20] J.L.V. Noguera, H.L. Ayala, C.E. Schaerer, M. Rolon, Mathematical morphology for counting trypanosoma cruzi amastigotes, in: 2013 XXXIX Latin American Computing Conference, CLEI, IEEE, 2013, pp. 1–12.
[21] J.L.V. Noguera, H.L. Ayala, C.E. Schaerer, J. Facon, A color morphological ordering method based on additive and subtractive spaces, in: 2014 IEEE International Conference on Image Processing, ICIP, IEEE, 2014, pp. 674–678.
[22] W. Banzhaf, P. Nordin, R.E. Keller, F.D. Francone, Genetic Programming: An Introduction, vol. 1, Morgan Kaufmann, San Francisco, 1998.
[23] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, vol. 1, MIT Press, 1992.
[24] W.B. Langdon, R. Poli, Foundations of Genetic Programming, Springer Science & Business Media, 2002.
[25] H. Kusetogullari, A. Yavariabdi, Evolutionary multiobjective multiple description wavelet based image coding in the presence of mixed noise in images, Appl. Soft Comput. 73 (2018) 1039–1052.
[26] A. Yavariabdi, H. Kusetogullari, Change detection in multispectral landsat images using multiobjective evolutionary algorithm, IEEE Geosci. Remote Sens. Lett. 14 (2017) 414–418.
[27] C.-W. Bong, M. Rajeswari, Multi-objective nature-inspired clustering and classification techniques for image segmentation, Appl. Soft Comput. 11 (2011) 3271–3282.
[28] M. Zhang, L. Jiao, W. Ma, J. Ma, M. Gong, Multi-objective evolutionary fuzzy clustering for image segmentation with moea/d, Appl. Soft Comput. 48 (2016) 621–637.
[29] A.A. Goshtasby, Similarity and dissimilarity measures, in: Image Registration, Springer, 2012, pp. 7–66.
[30] E. Senyurek, H. Polat, Effects of similarity measures on the quality of predictions, Gazi Univ. J. Sci. 26 (2014) 557–562.
[31] J.H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, University of Michigan Press, 1975.
[32] I. Kaya, A genetic algorithm approach to determine the sample size for attribute control charts, in: Including Special Issue on Artificial Imune Systems, Inf. Sci. 179 (2009) 1552–1566.
[33] C.H. Cheng, T.L. Chen, L.Y. Wei, A hybrid model based on rough sets theory and genetic algorithms for stock price forecasting, Inf. Sci. 180 (2010) 1610–1629.
[34] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE Trans. Evol. Comput. 6 (2002) 182–197.
[35] A. Abraham, N. Nedjah, L. de Macedo Mourelle, Evolutionary Computation: From Genetic Algorithms to Genetic Programming, Springer, 2006.
[36] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (1999) 257–271.
[37] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization, in: Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems, National Technical University of Athens & The University of Patras, 2002, pp. 1–12.
[38] J.J. Durillo, A.J. Nebro, jmetal: a Java framework for multi-objective optimization, Adv. Eng. Softw. 42 (2011) 760–771.
[39] D.E. Goldberg, et al., Genetic Algorithms in Search, Optimization, and Machine Learning, vol. 412, Addison-Wesley, Reading Menlo Park, 1989.
[40] J. MagalhāEs-Mendes, A comparative study of crossover operators for genetic algorithms to solve the job shop scheduling problem, WSEAS Trans. Comput. 12 (2013).
[41] J.M. Bader, Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods, CreateSpace, Paramount, CA, 2010.
[42] R.C. Gonzalez, R.E. Woods, et al., Digital Image Processing, 2002.